**Advanced Design System 2011**

**September 2011**
**System Models**

**Acknowledgments**

Mentor Graphics is a trademark of Mentor Graphics Corporation in the U.S. and other countries. Mentor products and processes are registered trademarks of Mentor Graphics Corporation. * Calibre is a trademark of Mentor Graphics Corporation in the US and other countries. "Microsoft®, Windows®, MS Windows®, Windows NT®, Windows 2000® and Windows Internet Explorer® are U.S. registered trademarks of Microsoft Corporation. Pentium® is a U.S. registered trademark of Intel Corporation. PostScript® and Acrobat® are trademarks of Adobe Systems Incorporated. UNIX® is a registered trademark of the Open Group. Oracle and Java and registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners. SystemC® is a registered trademark of Open SystemC Initiative, Inc. in the United States and other countries and is used with permission. MATLAB® is a U.S. registered trademark of The Math Works, Inc.. HiSIM2 source code, and all copyrights, trade secrets or other intellectual property rights in and to the source code in its entirety, is owned by Hiroshima University and STARC. FLEXlm is a trademark of Globetrotter Software, Incorporated. Layout Boolean Engine by Klaas Holwerda, v1.7 http://www.xs4all.nl/~kholwerd/bool.html . FreeType Project, Copyright (c) 1996-1999 by David Turner, Robert Wilhelm, and Werner Lemberg. QuestAgent search engine (c) 2000-2002, JObjects. Motif is a trademark of the Open Software Foundation. Netscape is a trademark of Netscape Communications Corporation. Netscape Portable Runtime (NSPR), Copyright (c) 1998-2003 The Mozilla Organization. A copy of the Mozilla Public License is at http://www.mozilla.org/MPL/ . FFTW, The Fastest Fourier Transform in the West, Copyright (c) 1997-1999 Massachusetts Institute of Technology. All rights reserved.

The following third-party libraries are used by the NlogN Momentum solver:

"This program includes Metis 4.0, Copyright © 1998, Regents of the University of Minnesota", http://www.cs.umn.edu/~metis , METIS was written by George Karypis (karypis@cs.umn.edu).

Intel@ Math Kernel Library, http://www.intel.com/software/products/mkl

SuperLU_MT version 2.0 - Copyright © 2003, The Regents of the University of California, through Lawrence Berkeley National Laboratory (subject to receipt of any required approvals from U.S. Dept. of Energy). All rights reserved. SuperLU Disclaimer: THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)

ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

7-zip - 7-Zip Copyright: Copyright (C) 1999-2009 Igor Pavlov. Licenses for files are: 7z.dll: GNU LGPL + unRAR restriction, All other files: GNU LGPL. 7-zip License: This library is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 2.1 of the License, or (at your option) any later version. This library is distributed in the hope that it will be useful,but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for more details. You should have received a copy of the GNU Lesser General Public License along with this library; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA. unRAR copyright: The decompression engine for RAR archives was developed using source code of unRAR program.All copyrights to original unRAR code are owned by Alexander Roshal. unRAR License: The unRAR sources cannot be used to re-create the RAR compression algorithm, which is proprietary. Distribution of modified unRAR sources in separate form or as a part of other software is permitted, provided that it is clearly stated in the documentation and source comments that the code may not be used to develop a RAR (WinRAR) compatible archiver. 7-zip Availability: http://www.7-zip.org/

AMD Version 2.2 - AMD Notice: The AMD code was modified. Used by permission. AMD copyright: AMD Version 2.2, Copyright © 2007 by Timothy A. Davis, Patrick R. Amestoy, and Iain S. Duff. All Rights Reserved. AMD License: Your use or distribution of AMD or any modified version of AMD implies that you agree to this License. This library is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 2.1 of the License, or (at your option) any later version. This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for more details. You should have received a copy of the GNU Lesser General Public License along with this library; if not, write to the Free Software Foundation, Inc., 51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA Permission is hereby granted to use or copy this program under the terms of the GNU LGPL, provided that the Copyright, this License, and the Availability of the original version is retained on all copies.User documentation of any code that uses this code or any modified version of this code must cite the Copyright, this License, the Availability note, and "Used by permission." Permission to modify the code and to distribute modified code is granted, provided the Copyright, this License, and the Availability note are retained, and a notice that the code was modified is included. AMD Availability: http://www.cise.ufl.edu/research/sparse/amd

UMFPACK 5.0.2 - UMFPACK Notice: The UMFPACK code was modified. Used by permission. UMFPACK Copyright: UMFPACK Copyright © 1995-2006 by Timothy A. Davis. All Rights Reserved. UMFPACK License: Your use or distribution of UMFPACK or any modified version of UMFPACK implies that you agree to this License. This library is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 2.1 of the License, or (at your option) any later version. This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for more details. You should have received a copy of the GNU Lesser General Public License

along with this library; if not, write to the Free Software Foundation, Inc., 51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA Permission is hereby granted to use or copy this program under the terms of the GNU LGPL, provided that the Copyright, this License, and the Availability of the original version is retained on all copies. User documentation of any code that uses this code or any modified version of this code must cite the Copyright, this License, the Availability note, and "Used by permission." Permission to modify the code and to distribute modified code is granted, provided the Copyright, this License, and the Availability note are retained, and a notice that the code was modified is included. UMFPACK Availability: http://www.cise.ufl.edu/research/sparse/umfpack  UMFPACK (including versions 2.2.1 and earlier, in FORTRAN) is available at http://www.cise.ufl.edu/research/sparse . MA38 is available in the Harwell Subroutine Library. This version of UMFPACK includes a modified form of COLAMD Version 2.0, originally released on Jan. 31, 2000, also available at http://www.cise.ufl.edu/research/sparse . COLAMD V2.0 is also incorporated as a built-in function in MATLAB version 6.1, by The MathWorks, Inc. http://www.mathworks.com . COLAMD V1.0 appears as a column-preordering in SuperLU (SuperLU is available at http://www.netlib.org ). UMFPACK v4.0 is a built-in routine in MATLAB 6.5. UMFPACK v4.3 is a built-in routine in MATLAB 7.1.

Qt Version 4.6.3 - Qt Notice: The Qt code was modified. Used by permission. Qt copyright: Qt Version 4.6.3, Copyright (c) 2010 by Nokia Corporation. All Rights Reserved. Qt License: Your use or distribution of Qt or any modified version of Qt implies that you agree to this License. This library is free software; you can redistribute it and/or modify it under the
terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 2.1 of the License, or (at your option) any later version. This library is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for more details. You should have received a copy of the GNU Lesser General Public License along with this library; if not, write to the Free Software Foundation, Inc., 51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA Permission is hereby granted to use or copy this program under the terms of the GNU LGPL, provided that the Copyright, this License, and the Availability of the original version is retained on all copies.User
documentation of any code that uses this code or any modified version of this code must cite the Copyright, this License, the Availability note, and "Used by permission." Permission to modify the code and to distribute modified code is granted, provided the Copyright, this License, and the Availability note are retained, and a notice that the code was modified is included. Qt Availability: http://www.qtsoftware.com/downloads  Patches Applied to Qt can be found in the installation at: $HPEESOF_DIR/prod/licenses/thirdparty/qt/patches. You may also contact Brian Buchanan at Agilent Inc. at brian_buchanan@agilent.com for more information.

The HiSIM_HV source code, and all copyrights, trade secrets or other intellectual property rights in and to the source code, is owned by Hiroshima University and/or STARC.

**Errata** The ADS product may contain references to "HP" or "HPEESOF" such as in file names and directory names. The business entity formerly known as "HP EEsof" is now part of Agilent Technologies and is known as "Agilent EEsof". To avoid broken functionality and to maintain backward compatibility for our customers, we did not change all the names and labels that contain "HP" or "HPEESOF" references.

**Warranty** The material contained in this document is provided "as is", and is subject to being changed, without notice, in future editions. Further, to the maximum extent permitted by applicable law, Agilent disclaims all warranties, either express or implied, with regard to this documentation and any information contained herein, including but not limited to the implied warranties of merchantability and fitness for a particular purpose. Agilent shall not be liable for errors or for incidental or consequential damages in connection with the furnishing, use, or performance of this document or of any information contained herein. Should Agilent and the user have a separate written agreement with warranty terms covering the material in this document that conflict with these terms, the warranty terms in the separate agreement shall control.

**Technology Licenses** The hardware and/or software described in this document are furnished under a license and may be used or copied only in accordance with the terms of such license. Portions of this product include the SystemC software licensed under Open Source terms, which are available for download at http://systemc.org/ . This software is redistributed by Agilent. The Contributors of the SystemC software provide this software "as is" and offer no warranty of any kind, express or implied, including without limitation warranties or conditions or title and non-infringement, and implied warranties or conditions merchantability and fitness for a particular purpose. Contributors shall not be liable for any damages of any kind including without limitation direct, indirect, special, incidental and consequential damages, such as lost profits. Any provisions that differ from this disclaimer are offered by Agilent only.

**Restricted Rights Legend** U.S. Government Restricted Rights. Software and technical data rights granted to the federal government include only those rights customarily provided to end user customers. Agilent provides this customary commercial license in Software and technical data pursuant to FAR 12.211 (Technical Data) and 12.212 (Computer Software) and, for the Department of Defense, DFARS 252.227-7015 (Technical Data - Commercial Items) and DFARS 227.7202-3 (Rights in Commercial Computer Software or Computer Software Documentation).

# Amplifiers and Mixers

## Introduction

The *Filters - <filter type>* and *System - <device type>* palettes contain two fundamentally different types of behavioral system models.

*Filters*, *System - Amps & Mixers*, and *System - Mod/Demod* can be classified as *tops-down system models* that support a tops-down system design flow where model behaviors are characterized by a small number of independent parameters such as frequency, power and load. They are often referred to as *parameter-based behavioral models*.

*System - Data Models* can be classified as *bottoms-up system models* that support a bottoms-up verification flow where model behaviors are extracted from a simulation (or measurement) of a transistor-level circuit. They are often referred to as *data-based behavioral models*.

The *parameter-based behavioral models* typically provide superior speed relative to the data-based behavioral models with both of these being vastly superior to a brute-force transistor-level simulation.

The *data-based behavioral models* typically provide superior accuracy relative to the *parameter-based behavioral models* as they capture actual behaviors of implemented circuit components and not just design specifications.

The differences between *parameter-* and *data-based behavioral models* justify a palette emphasis on flow (all data-based behavioral models grouped together) rather than functionality (all amplifiers, mixers, modulators, and demodulators grouped together) and resulted in the addition of a *System - Data Models* palette.

The use model for parameter-based behavioral models is to simply set a series of parameters prior to using the model. The use model for data-based behavioral models is slightly more involved. For a discussion, see *System Data Models* (ccsys).

## Curve-Fitting Algorithm

The curve-fitting algorithm to determine the nonlinear behavior of the system mixer models is based on fitting a polynomial to the specified data where the saturation power (Psat) is calculated when the derivative of this polynomial is zero.

$$Pn(x) = a1*x1+a2*x2^2+a3*x^3+...$$

It is important to note that the coefficients a4,a6,a8,...are always zero. In only one case a2 is non-zero and that's when SOI and TOI are specified.

| Parameters | Order |
|---|---|
| TOI | 3 |
| TOI & AM2PM | 3 |
| SOI & TOI | 3 a2 ≠ 0 |
| PndB | 3 |
| PndB & AM2PM | 3 |
| Psat | 5 |
| TOI & PndB | 5 |
| PndB & Psat | 7 |
| TOI & Psat | 7 |
| PndB & TOI & Psat | 9 |

## Components

- *AGC Amp (Voltage-Controlled Amplifier for AGC loops)* (ccsys)
- *AGC PwrControl (Power Control Block for AGC loops)* (ccsys)
- *Amplifier2 (RF System Amplifier)* (ccsys)
- *Amplifier (Obsolete RF System Amplifier)* (ccsys)
- *AmplifierVC (Ideal Voltage-Controlled Amplifier)* (ccsys)
- *AmpSingleCarrier (Single Carrier Amplifier)* (ccsys)
- *FreqMult (Ideal Frequency Multiplier)* (ccsys)
- *LogACDemod (Demodulating AC Logarithmic Amplifier)* (ccsys)
- *LogDC (DC Logarithmic Amplifier)* (ccsys)
- *LogSuccDetect (Successive Detection Logarithmic Amplifier* (ccsys)
- *LogTrue (True Logarithmic Amplifier)* (ccsys)
- *Mixer2 (RF System Mixer)* (ccsys)
- *Mixer (First RF System Mixer, Polynomial Model for Nonlinearity)* (ccsys)
- *MixerWithLO (Mixer with Internal Local Oscillator)* (ccsys)
- *OpAmp (Operational Amplifier)* (ccsys)
- *OpAmpIdeal (Ideal Operational Amplifier)* (ccsys)
- *VMult (Voltage Multiplier)* (ccsys)

# AGC_Amp (Voltage-Controlled Amplifier for AGC loops)

## Symbol
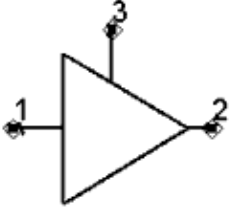


## Parameters

| Name | Description | Units | Default |
|---|---|---|---|
| ZRef | Reference resistance for all ports | Ohm | 50 |
| S11 | Forward reflection coefficient, use x + j × y, polar(x,y), dbpolar(x,y), vswrpolar(x,y) for complex value | None | polar(0,0) |
| S22 | Reverse reflection coefficient, use x + j × y, polar(x,y), dbpolar(x,y), vswrpolar(x,y) for complex value | None | polar(0,180) |
| Min_dB | Minimum gain; occurs when _v3 = Min_dB volt | dB | −40 |
| Max_dB | Maximum gain; occurs when _v3 = Max_dB volt | dB | 40 |
| NF | Noise Figure [NF mode used for NFmin=0] | dB | 0 |
| SOI | Second Order Intercept | dBm | 1000 |
| TOI | Third Order Intercept | dBm | 1000 |

### Range of Usage

ZRef > 0
| Sij | > 0 (ij=11; 22)
Min_dB < Max_dB
NF ≥ 0 dB

### Notes/Equations

1. AGC_Amp is a voltage-controlled amplifier with user-defined input and output reflection coefficients, minimum and maximum gain limits, and nonlinearity.
2. This component uses internal components TwoPort, Noisy2Port, AmplifierVC, and Amplifier2, as shown in the schematic.
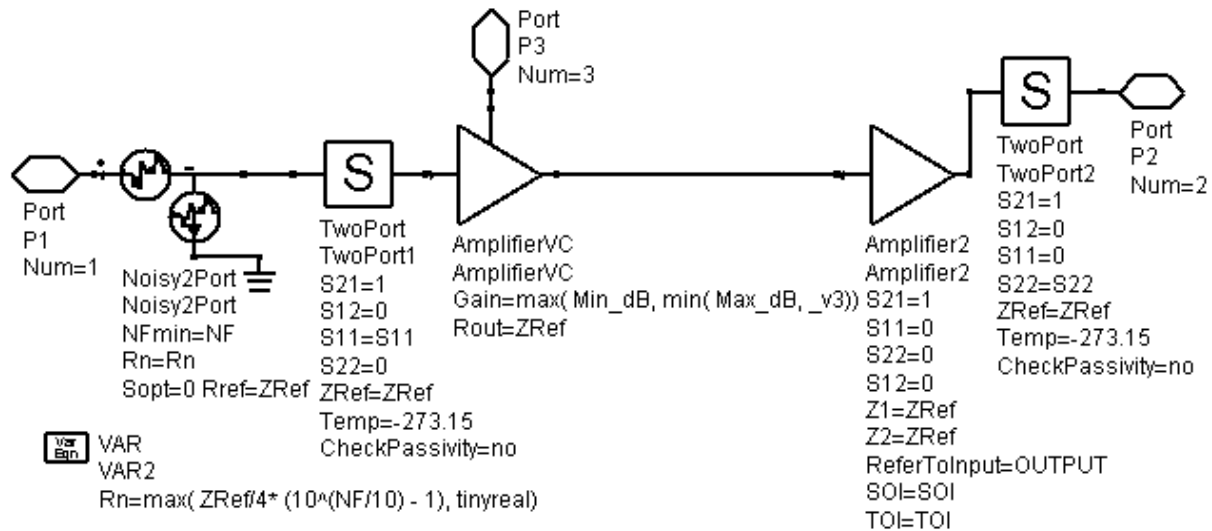
3. Control voltage at pin 3 is used to set gain. Under small signal conditions, gain in dB as a function of the control voltage at port 3: _v3 is:
   Gain_dB = max( Min_dB, min( Max_dB, _v3))
4. The input impedance at pin 3 is infinite.
5. Regarding NF, component noise parameters (NFmin, Rn, Sopt) are related to NF by setting NFmin=NF, Sopt=0, and Rn=max( ZRef/4 × ($10^{NF/10}$-1), tinyreal), where tinyreal is an internal simulator value for the smallest real value allowed.
6. Regarding SOI and TOI, AGC_Amp must be output-matched in order to validate SOI and TOI.
   By default, SOI and TOI are blank resulting in no nonlinearity. TOI can be set without setting SOI; if SOI is set, TOI must be set.
   For more information about SOI and TOI, refer to Amplifier2 documentation.
7. AGC_Amp is required for defining AGC control loops in designs to be simulated with Budget Controller. Such Budget simulation requires AGC_PwrControl; refer to AGC_PwrControl for AGC control loop usage details.
8. Example designs demonstrating AGC_Amp in AGC control loops are located in ADS; access the designs from the ADS Main window: **File** > **Open** > **Example** > **Tutorial** > *RF_Budget_Examples_wrk* ; see *Budget_AGC* and *AGC_loop_CE_test* .

# AGC_PwrControl (Power Control Block for AGC loops)

## Symbol
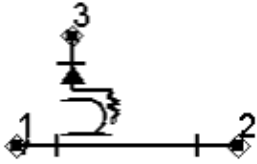


## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| ZRef | Reference impedance for ports 1 and 2 | Ohm | 50 |
| S11 | Port 1 reflection coefficient, complex number | None | polar(0,0) |
| S22 | Port 2 reflection coefficient, complex number | None | polar(0,180) |
| Fnom | Nominal input frequency for detection and power control | GHz | 1 |
| TargetPwr | Target power output level | dBm | 10 |
| BW | Closed loop −3dB frequency | kHz | 4.5 |
| DampingFactor | Loop damping factor | None | 0.707 |
| NormalizedZero | Loop normalized zero | None | 1 |
| ExternalGain | Open loop gain (voltage ratio) external to this block | None | 0.025 |

## Range of Usage

ZRef > 0
| Sij | > 0 (ij=11; 22)
Fnom > 0
TargetPower > 0 (Watts)
BW > 0
DampingFactor > 0
NormalizedZero > 0
ExternalGain > 0

## Notes/Equations

1. AGC_PwrControl is used with AGC control loops; it incorporates an OpAmp loop filter with output at pin 3 for driving a preceeding AGC amplifier toward the gain required to achieve the TargetPwr level at the AGC_PwrControl input.
2. This component uses internal components IQ_DemodTuned, FDD (used as a log amp), and OpAmp active filter, as shown in the schematic.
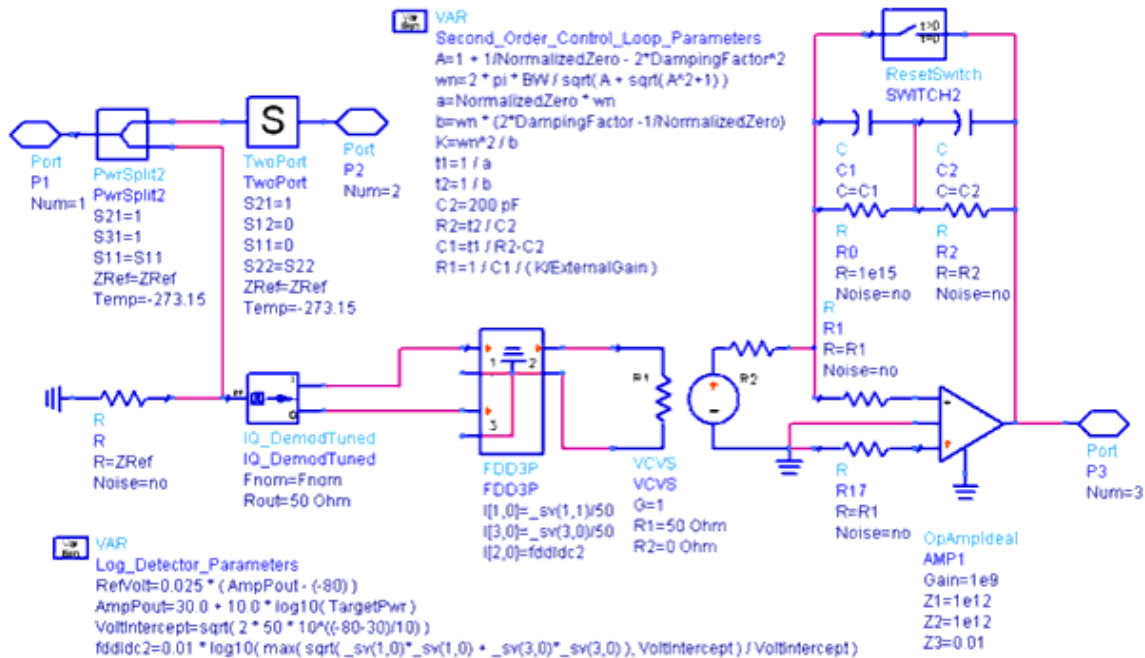
3. The control voltage at pin 3 is used for feedback to an AGC amplifier (pin 3 of AGC_Amp) to achieve an RF amplitude control loop.
4. AGC_PwrControl depends on time-varying RF waveforms, so it is typically used for Envelope simulations when an input modulated RF signal exists at Fnom.
5. For Budget simulations, AGC_PwrControl is used with AGC_Amp to define AGC loops. Budget simulation AGC control loops work with one tone and only the steady state value of the loop is used.
   For Budget simulations, Fnom, BW, DampingFactor, NormalizedZero, and ExternalGain parameters are not used.
6. The time-domain RF control loop dehavior is defined for second-order control loop operation in which the OpAmp loop filter internal to this model establishes the second-order control loop filter characteristics for the combined circuit external to AGC_PwrControl plus AGC_PwrControl.
7. The control loop drives the external AGC amplifier to a gain level that is sufficient for maintaining the output power of the AGC system at the user-specified value of TargetPwr.
8. Loop dynamics follow second-order control loop theory whereby the user defines TargetPwr, -3 dB closed loop bandwidth (BW), loop gain external to AGC_PwrControl, and second-order loop parameters (NormalizedZero and DampingFactor) and AGC_PwrControl sets the internal OpAmp reference voltage (RefVolt) and loop filter capacitance and resistance values (R1, R2, C1, C2). Refer to the schematic for the defining equations. AGC_PwrControl uses a log detector defined by use of an FDD model.
9. For HarmonicBalance use, this model behavior defaults with output of a DC signal voltage at pin 3 equal to the reference voltage at the non-inverting input of the internal OpAmp.
10. The output impedance at pin 3 is zero.
11. For details regarding second-order loop theory, refer to the Second-Order Control Loop Filters section in chapter 2 of *Timed Components* (timed) documentation.
12. Example designs demonstrating AGC_PwrControl in AGC control loops are located in ADS; access the designs from the ADS Main window: **File** > **Open** > **Example** > **Tutorial** > *RF_Budget_Examples_wrk* ; see *Budget_AGC* and *AGC_loop_CE_test* .

# Amplifier2 (RF System Amplifier)

## Symbol



## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| S21 | Forward transmission coefficient, use x + j × y, polar(x,y), dbpolar(x,y) for complex value | None | dbpolar(0,0) |
| S11 | Forward reflection coefficient, use x + j × y, polar(x,y), dbpolar(x,y), vswrpolar(x,y) for complex value | None | polar(0,0) |
| S22 | Reverse Reflection Coefficient, use x + j × y, polar(x,y), dbpolar(x,y), vswrpolar(x,y) for complex value | None | polar(0,180) |
| S12 | Reverse Transmission Coefficient, use x + j × y, polar(x,y), dbpolar(x,y) for complex value | None | 0 |
| NF | Noise figure [NF mode used for NFmin=0] | dB | None |
| NFmin | Minimum noise figure at Sopt [(NFmin,Sopt,Rn) mode used for NFmin>0] | dB | None |
| Sopt | Optimum source reflection for minimum noise figure [(NFmin,Sopt,Rn) mode used for NFmin>0] | None | None |
| Rn | Equivalent noise resistance [(NFmin,Sopt,Rn) mode used for NFmin>0] | None | |
| Z1 | Reference impedance for port 1 (must be a real number) | None | |
| Z2 | Reference impedance for port 2 (must be a real number) | None | |
| GainCompType | Gain compression type | None | LIST |
| GainCompFreq | Frequency at which Gain Compression is specified | None | |
| ReferToInput | Specify each of the gain compression options outlined in *Polynomial Order for Various Magnitude Modes* with respect to the input or output power of the device | None | OUTPUT |
| SOI | Second order intercept | dBm | None |
| TOI | Third order intercept | dBm | None |
| Psat | Power saturation point (always referred to output, regardless of the value of the ReferToInput parameter) | dBm | None |
| GainCompSat | Gain compression at Psat | dB | 5.0 |
| GainCompPower | Power Level at gain compression specified by GainComp | dBm | None |
| GainComp | Gain Compression at GainCompPower | dB | 1.0 |
| AM2PM | Amplitude modulation to phase modulation | deg/dB | None |
| PAM2PM | Power level at AM2PM | dBm | None |
| GainCompFile | Filename for gain compression data in S2D file format | None | None |
| ClipDataFile | Clip data beyond maximum input power: YES=enable, NO=disable | None | yes |
| ImpNoncausalLength | Non-causal function impulse response order | Integer | None |
| ImpMode | Convolution mode | Integer | None |
| ImpMaxFreq | Maximum frequency to which device is evaluated | None | |
| ImpDeltaFreq | Sample spacing in frequency | None | |
| ImpMaxOrder | Maximum allowed impulse response order | Integer | None |
| ImpWindow | Smoothing window | Integer | None |
| ImpRelTol | Relative impulse response truncation factor | None | None |
| ImpAbsTol | Absolute impulse response truncation factor | None | None |

## Frequently Asked Questions

**Q1** : What are the major differences between Amplifier and Amplifier2?
**A1** : Refer to *note 1*.

**Q2** : What are the supported parameter combinations?
**A2** : Refer to Range of Usage.

**Q3** : What is the range of usage for each parameter combination?
**A3** : Refer to Range of Usage.

**Q4** : What model is being used?
**A4** : A polynomial model. Refer to Modeling Basics.

**Q5** : What polynomial order is being used?
**A5** : It depends on the parameters set on the component. Refer to Modeling Basics.

**Q6** : Can these polynomials blow up?
**A6** : No, these polynomials are limited. Refer to Modeling Basics for details.

**Q7** : The parameters specified for Amplifier2 match those for my transistor level amplifier. Why don't the harmonics generated by Amplifier2 match those for my transistor level amplifier?
**A7** : Amplifier2 only matches the specified parameters. Refer to Modeling Basics for details.

**Q8** : I have a small-signal and a large-signal tone. Why does the small-signal tone have a smaller gain than the large-signal tone?
**A8** : It is a consequence of the polynomial model adopted by Amplifier2. Refer to Modeling Basics for details.

**Q9** : In a swept-power harmonic halance simulation, my higher-order harmonics are zero up to a certain order but then they suddenly experience a very sharp increase. Isn't this wrong?
**A9** : No, it's called generating a square wave via hard limiting. Refer to Modeling Basics for details.

**Q10** : How do I validate AM to PM conversion?
**A10** : Refer to AM to PM Conversion.

**Q11** : Why does AM to PM conversion only match my transistor level amplifier around the power level PAM2PM at which I specified AM2PM?
**A11** : With information at only one power level, Amplifier2 has to rely on assumptions and generic curve shapes to model AM to PM at all power levels. For better modeling, use an S2D file and specify the exact magnitude and phase variation of your amplifier. Refer to AM to PM Conversion for details.

**Q12** : With AM to PM conversion enabled, my output phase is supposed to be constant at high power levels. Why does my AM to PM start changing at high power levels?
**A12** : This is probably due to harmonic balance aliasing errors and can be mitigated by oversampling. Refer to AM to PM Conversion for details.

**Q13** : With AM to PM conversion enabled, why don't the predicted second- and third-order intercept points match SOI and TOI as set on Amplifier2?
**A13** : It is a consequence of fitting the magnitude and phase separately. Refer to Refer to AM to PM Conversion for details.

**Q14** : Why do Amplifier2's noise results differ from those for Amplifier?
**A14** : Amplifier2 models noise differently from Amplifier. Refer to [Noise](#) for details.

**Q15** : What noise model is used by Amplifier2?
**A15** : Refer to [Noise](#).

**Q16** : What is "NF-only mode" and "(NFmin,Sopt,Rn) mode" for Amplifier2?
**A16** : It is two different ways of specifying noise. Refer to [Noise](#) for details.

**Q17** : How is NFssb/NFdsb calculated, both at low powers and in compression?
**A17** : Refer to [Noise](#).

**Q18** : Can you provide more details about the noise voltages and noise figures produced by Amplifier2 for the different modes of operation?
**A18** : Refer to [Noise](#) .

**Q19** : With complex S21, why does Amplifier2 differ from Amplifier?
**A19** : Amplifier2 handles complex S21 values differently from Amplifier, leading to more physical waveforms. Refer to *note 5* for details.

**Q20** : Why doesn't Amplifier2 work for complex reference impedances?
**A20** : Amplifier2 does not support complex reference impedances, per *note 7*.

**Q21** : Why is Amplifier2 sometimes slower than Amplifier?
**A21** : This is a consequence of the implementational differences between Amplifier and Amplifier2. Refer to *note 10* for details.

**Q22** : Why don't the predicted second- and third-order intercept points match SOI and TOI as set on Amplifier2?
**A22** : You are probably not setting up your validation correctly. Make sure to do a two-tone simulation, not a one-tone simulation. Refer to *note 11* for details.

**Q23** : Why does my ACPR level get better as I get deeper into compression?
**A23** : This is a consequence of the polynomial model adopted by Amplifier2. Refer to *note 12* for details.

**Q24** : Why does Amplifier2 ignore the ACDATA block of my S2D file?
**A24** : It uses the S-parameters on the component instead. Refer to *note 13* for details.

**Q25** : When the power range in an S2D file differs from that of a simulation, which power range is used for polynomial fitting?
**A25** : The power range in the S2D file. Refer to *note 15* for details.

**Q26** : Why do I get ringing at high powers when using an S2D file?
**A26** : This is a consequence of the polynomial model adopted by Amplifier2. Refer to *note 16* for details.

**Q27** : How do I get rid of this ringing?
**A27** : Eliminate data for high input powers, rely on extrapolation via ClipDataFile=yes, or break the S2D file into two. Worst case, use AmpSingleCarrier. Refer to *note 16* for details.

**Q28** : Why do I see different results when I use the Analog/RF Amplifier2 component and the Ptolemy GainRF component with the same parameters?

**A28** : While based on the same OmniSys legacy, the implementations differ and thus the components can give different results. Refer to *note 18* for details.

### Range of Usage

|Sij| > 0 (i=1,2; j=1,2)
NF ≥ 0 dB
NFmin ≥ 0 dB
0 < |Sopt| < 1
Rn > 0
GainCompFreq > 0
0 dB < GainComp < 3 dB

When specifying gain compression using model parameters, only certain combinations of parameters will produce stable polynomial curve fitting. If unrealistic parameter values are used, the polynomial will become unstable, resulting in oscillations. The recommended parameter combinations are listed here:

- Third-order intercept parameter:
  Parameters: TOI
  Range of validity: N/A
- Gain compression parameters:
  Parameters: GainCompPower, GainComp
  Range of validity: N/A
- Power saturation parameters:
  Parameters: Psat, GainCompSat
  Range of validity: N/A
- Third-order intercept and 1dB gain compression parameters:
  Parameters: TOI, GainCompPower with GainComp=1dB
  Range of validity: TOI > GainCompPower + 10.8
- Third-order intercept and power saturation parameters:
  Parameters: TOI, Psat, GainCompSat
  Range of validity: TOI > Psat + 8.6
- 1dB gain compression and power saturation parameters:
  Parameters: GainCompPower with GainComp=1dB, Psat, GainCompSat
  Range of validity: Psat > GainCompPower + 3
- Third-order intercept, 1dB gain compression and power saturation parameters:
  Parameters: TOI, GainCompPower with GainComp=1dB, Psat, GainCompSat
  Range of validity: Psat > GainCompPower +3, TOI > GainCompPower + 10.8
- Second-order intercept and third-order intercept parameters:
  Parameters: SOI, TOI
  Range of validity: N/A

### Modeling Basics

Amplifier2 is based on a polynomial model of the magnitude of the output voltage as a function of the input voltage. If SOI is not specified, the magnitude response is modeled

using a polynomial of odd orders

$y = a1 \times x + a3 \times x^3 + a5 \times x^5 + \ldots$

The order of the polynomial depends on the data entered by the user. If SOI is specified, the magnitude response has an even order term

$y = a1 \times x + a2 \times x^2 + a3 \times x^3$

The order of the polynomial is hardwired at 3. The polynomial orders for the various magnitude modes are summarized in *Polynomial Order for Various Magnitude Modes*.

| Magnitude Mode | Polynomial Order |
|---|---|
| TOI | 3 |
| SOI, TOI | 3 (with second-order term) |
| GainCompPower, GainComp | 3 |
| Psat, GainCompSat | 5 |
| TOI, GainComp=1dB, GainCompPower | 5 |
| TOI, Psat, GainCompSat | 7 |
| GainComp=1dB, GainCompPower, Psat, GainCompSat | 7 |
| TOI, GainComp=1dB, GainCompPower, Psat, GainCompSat | 9 |

For GainCompType=LIST, the phase response is described in the section AM to PM Conversion. For GainCompType=FILE and a GCOMP1-GCOMP6 block in the S2D file, the phase response is zero. For GainCompType=FILE and a GCOMP7 block in the S2D file, the phase response is modeled using an odd order polynomial, just like the magnitude response.

To prevent these polynomial expressions from blowing up and resulting in a non-physical model, the polynomial model is only used up to the largest $x$ fulfilling $dy/dx=0$, denoted by $x0$. Above this value, the amplifier is hard-limited at $y(x0)$.

The polynomial coefficients for Amplifier2 are based on the parameters set for Amplifier2 regardless of the context in which Amplifier2 is used and the actual number of tones at the input of Amplifier2. These polynomial coefficients are then applied when Amplifier2 is analyzed in the environment where it lives. For example, given GainComp and GainCompPower, the polynomial coefficients are determined by exciting Amplifier2 with a one-tone signal and requiring the compressed output power to be GainComp below the linear response at GainCompPower. This can be confirmed by doing a one-tone analysis of Amplifier2. These polynomial coefficients are then used in any analysis using Amplifier2, whether it be a one-tone or a multi-tone analysis.

Amplifier2 matches the parameters that have been set on the component but is not guaranteed to match any other characteristics of a transistor level amplifier. Assume three parameters have been specified for Amplifier2: the linear gain S21, the 1 dB compression point GainCompPower (GainComp is set to 1 dB), and the third-order intercept TOI. This means that the behavior of Amplifier2 must match these parameters and nothing else.

This is achieved by modeling $y$ as a function of $x$ via $y = a1 \times x + a3 \times x^3 + a5 \times x^5$. The three known quantities allow the determination of the three unknown polynomial coefficients. If we look at this equation, we see that there is a fifth-order term. Therefore, Amplifier2 will produce fifth-order harmonics. However, this fifth-order order term is not supposed to match that from the transistor level amplifier from which the three parameters S21, GainCompPower and TOI were extracted. No information about the fifth-order intercept of that amplifier has been specified so we cannot match it. The fifth-order

term for Amplifier2 is necessary to properly model S21, GainCompPower and TOI and these parameters are properly modeled. In general, a transistor level amplifier will produce finite higher order harmonics which Amplifier2 does not produce.

For GainCompType=FILE and a GCOMP7 block in the S2D file, the magnitude and phase data in the GCOMP7 block of the S2D file is fitted to two separate polynomials. With n power points in the GCOMP7 block, the polynomial order will be min(2 × n -1,9). In most practical cases, the GCOMP7 block will have data at five or more power points and consequently ninth-order polynomials with odd-only terms will be used.

Consider now an amplifier with the response

$$y = a1 \times x + a2 \times x^2 + a3 \times x^3$$

This is the model used when S21, SOI and TOI are specified. If $a2$ and $a3$ are non-zero, they will be negative. When $a2=0$, this is the model used when S21 and TOI are specified. When $a2=a3=0$, this represents a linear amplifier. The behavior of this amplifier can be analyzed analytically when the excitation is simple enough. Doing so gives a good understanding of the behavior of more complicated amplifiers with different excitations.

Assuming the excitation

$$x(t) = A1 \times \cos(\omega1 \times t)$$

we get the response

$$y(t) = \frac{1}{2} \times a2 \times A1^2$$

$$+ \left(a1 \times A1 + \frac{3}{4} \times a3 \times A1^3\right) \times \cos(\omega1 \times t)$$

$$+ \frac{1}{2} \times a2 \times A1^2 \times \cos(2 \times \omega1 \times t)$$

$$+ \frac{1}{4} \times a3 \times A1^3 \times \cos(3 \times \omega1 \times t)$$

and can define the gain of the fundamental A1 × cos(ω1 × t) as

$$G1 = a1 + \frac{3}{4} \times a3 \times A1^2$$

Assuming the excitation

$$x(t) = A1 \times \cos(\omega1 \times t) + A2 \times \cos(\omega2 \times t)$$

we get the response

$$y(t) = \frac{1}{2} \times a2 \times (A1^2 + A2^2)$$

$$+ \left(a1 \times A1 + \frac{3}{4} \times a3 \times A1^3 + \frac{3}{2} \times a3 \times A1 \times A2^2\right) \times \cos(\omega1 \times t)$$

$$+ \left(a1 \times A2 + \frac{3}{4} \times a3 \times A2^3 + \frac{3}{2} \times a3 \times A2 \times A1^2\right) \times \cos(\omega2 \times t)$$

$$+ \frac{1}{2} \times a2 \times A1^2 \times \cos(2 \times \omega1 \times t)$$

$$+ \frac{1}{2} \times a2 \times A2^2 \times \cos(2 \times \omega2 \times t)$$

$$+ \frac{1}{4} \times a3 \times A1^3 \times \cos(3 \times \omega1 \times t)$$

$$+ \frac{1}{4} \times a3 \times A2^3 \times \cos(3 \times \omega2 \times t)$$

$$+ a2 \times A1 \times A2 \times \cos((\omega1 - \omega2) \times t)$$

$$+ a2 \times A1 \times A2 \times \cos((\omega1 + \omega2) \times t)$$

$$+ \frac{3}{4} \times a3 \times A2 \times A1^2 \times \cos((2 \times \omega1 + \omega2) \times t)$$

$$+ \frac{3}{4} \times a3 \times A2 \times A1^2 \times \cos((2 \times \omega1 - \omega2) \times t)$$

$$+ \frac{3}{4} \times a3 \times A1 \times A2^2 \times \cos((2 \times \omega2 + \omega1) \times t)$$

$$+ \frac{3}{4} \times a3 \times A1 \times A2^2 \times \cos((2 \times \omega2 - \omega1) \times t)$$

and can define the gain of the fundamentals $A1 \times \cos(\omega1 \times t)$ and $A2 \times \cos(\omega2 \times t)$ as

$$G1 = a1 + a3 \times \left( \frac{3}{4} \times A1^2 + \frac{3}{2} \times A2^2 \right)$$

and

$$G2 = a1 + a3 \times \left( \frac{3}{2} \times A1^2 + \frac{3}{4} \times A2^2 \right)$$

respectively.

Examining $G1$ and $G2$ closer, we see that $G1>G2$ for $A1>A2$ and $a3<0$. This means that if we excite a polynomial amplifier with a large- and a small-signal tone, the large-signal tone will experience a larger gain than the small-signal tone. For more details, please refer to "Polynomial Model of Blocker Effects on LNA/Mixer Devices" by W. Domino et. al. in the June 2001 issue of Applied Microwave and Wireless.

This behavior is counter-intuitive for some people. If we excite an amplifier with a small-signal tone, it will provide its maximum/linear gain. If we excite the same amplifier with a large-signal tone, it will provide a smaller/compressed gain. If we excite an amplifier with both a large- and a small-signal tone, one could then expect that the small-signal tone would be subject to a larger gain than the large-signal tone. The above shows that this is not how Amplifier2 behaves.

This behavior, however, is intuitive for some people. If we consider an ideal hard limiting amplifier with an input voltage of Vin and an output voltage of Vmax, it has a large-signal gain of Vmax/Vin but a small-signal gain of zero, suggesting a larger large-signal gain than small-signal gain. This is in line with how Amplifier2 behaves.

Above, we presented the polynomial model used for modeling the output voltage as a function of the input voltage. This polynomial model, however, does not fully describe the response of the amplifier. It is important to remember that the amplifier has a power level at which it saturates and above which the output is clipped. We also presented the result of pushing A1 × cos(ω1 × t) and A1 × cos(ω1 × t)+A2 × cos(ω2 × t) signals through a $y$ = $a1 \times x + a2 \times x^2 + a3 \times x^3$ nonlinearity. The result is an output voltage consisting of a finite number of $C_n \times \cos( n \times \omega_n \times t )$ terms. More generally, it can easily be shown that pushing a sum of $A_i \times \cos( \omega_i \times t )$ terms through a polynomial nonlinearity will result in a finite number of $C_n \times \cos( n \times \omega_n \times t )$ terms. It might then be expected that a harmonic balance analysis of such an amplifier can only produce finite harmonics up to a certain order but will always produce zero harmonics beyond a certain order. This is the case when the amplifier is not hard limiting but is not the case when the amplifier starts hard limiting.

When we analyze an amplifier via a harmonic balance simulation of a certain order, the harmonics we will see on the input and output of the amplifier will be those which lead to the most correct input and output waveforms that can be achieved with the given order.

Assume the input waveform is a sine wave. This can be represented exactly using only the fundamental. Therefore, we will see a finite fundamental tone and zero harmonics on the input.

The output waveform depends on the input power level.

At low input powers, the output waveform is simply a sine wave, namely the input sine wave scaled by the linear amplifier gain. This can be represented exactly using only the fundamental. Therefore, we will see a finite fundamental tone and zero harmonics.

At higher input powers where the amplifier enters compression, the output waveform is the input sine wave pushed through the amplifier polynomial. Because a sine wave pushed through a polynomial gives rise to sine terms and no others, the output waveform will have finite harmonics up to a certain order and zero harmonics above that order.

At high enough input powers that the amplifier starts hard limiting, the output waveform is a clipped sine wave. For increasingly high input powers, this clipped sine wave approaches a square wave. Representation of such a clipped wave requires finite third, fifth, seventh, etc. harmonics so when the amplifier starts clipping all harmonics will be finite. This means that we will see a very sharp increase in the level of all such harmonics as soon as the amplifier starts clipping. If we increase the input power enough, we get approximately a square wave output. When this happens, the levels of the various harmonics reach saturation. The relative values of these saturated power levels can be predicted from the Fourier representation of a square wave.

### AM to PM Conversion

The Amplifier component supports AM to PM conversion for a very limited number of magnitude modes; Amplifier2 supports AM to PM conversion for all magnitude modes when GainCompType=LIST. Magnitude and phase is fitted separately so AM to PM conversion can be added for each magnitude mode. This adds a phase response but does

not alter the magnitude response.

AM to PM conversion is defined as the amount of phase change in degrees per magnitude/power change in dB. It is specified via the two parameters AM2PM and PAM2PM. AM2PM [degrees/dB] is the amount of AM to PM conversion while PAM2PM [dBm] is the power level at which the amplifier has this amount of AM to PM conversion. A physically sensible phase response is then constructed which has a phase of phase(S21) at low powers, an amount of AM to PM conversion given by AM2PM at PAM2PM, constant phase at high powers and for which the maximum amount of AM to PM at any power level is also given by AM2PM. Stated differently, the derivative of the phase response with respect to power in dB takes its maximum value AM2PM at PAM2PM.

This phase response is far from unique and certainly will not be correct for all amplifiers. Based only on the amount of AM to PM at one power level, there is simply no way to construct a phase response that matches any circuit level amplifier at all power levels. This AM to PM capability is only meant as a crude way of incorporating a phase response for amplifiers about which very little information is known - at the initial system level design iterations, For example, where no transistor level amplifier has been designed. Once more information about the amplifier's phase response is known and a more accurate phase modeling is desired, use Amplifier2 with GainCompType=FILE and specify the (compression and) phase response as a function of power in the GCOMP7 block of an S2D file. This is much more accurate than what this AM to PM capability can provide.

To document the exact phase response in a little more detail, we start out with AM2PM [degrees/dB] and PAM2PM [dBm]. We first define:

$k$=AM2PM

$P=10^{(PAM2PM-30)/10}$

For a sine wave

$A0 \times \cos(\omega \times t)$

the average power is

$$P = \int_{period} (A0 \times \cos(\omega \times t))^2 dt = \frac{A0^2}{2}$$

Therefore, we have

$$A0 = \sqrt{2 \times P}$$

The phase response is initially modeled in the A-domain. We choose

$y(A)=c0+c1 \times A+c2 \times A^2+c3 \times A^3$

We now get

$y(0)=0$ -> $c0=0$

$dy/dA(0)=0$ -> $c1=0$

$dy/dA(A0)= k$ -> $2 \times A0 \times c2+3 \times A0^2 \times c3=0$

$d^2 y / dA^2 (A0)=0$ -> $c2+3 \times A0 \times c3=0$

$c2=k/A0$

$c3=- k /(3 \times A0^2 )$

Then, we have

$y(A)= k / A0 \times A^2 - k /(3 \times A0^2 ) \times A^3$

This function has the following properties

$y(0)=0$

$dy / dA(0)=0$

$y (A0)=2/3 \times k \times A0$

$dy / dA(A0)=k$

$d^2 y / dA^2$ ($A0$)=0 ($A0$ is an extremum, max AM to PM occurs here)
$y$ ($2 \times A0$)=$4/3 \times k \times A0$=$2 \times y$ ($A0$)
$dy / dA$ ($2 \times A0$)=0

This curve will be distorted when it is made a function of power in dBm instead of magnitude $A$. The difference between two log values can be quite significant at small values but will be much smaller at large values. This means that the log function will distort the curve - it will be stretched out at low powers and will be compressed at high powers. This will change the point where the maximum slope occurs so the curve is shifted/scaled such that the maximum slope AM2PM occurs at PAM2PM. Also, we account for the fact that the derivative with respect to A must be in dB and not linear numbers.

The above phase response applies to the fundamental tone. This phase shift is applied before the application of the nonlinear polynomial which means that the change of the phase response of the *n*th order harmonic from its low power value will equal that for the fundamental tone scaled by the harmonic index *n*.

The easiest way to validate that the amount of AM to PM conversion takes its maximum value AM2PM at PAM2PM is to use the *diff* function in ADS. If the Amplifier2 output voltage is denoted by Vout and determined via a swept input power harmonic balance simulation, simply plot diff(phase(Vout[::,1])). The peak of this curve should occur at PAM2PM and should take the value AM2PM.

All harmonic balance simulations can be subject to aliasing errors. This can significantly change the phase response of an amplifier. To mitigate the effects of harmonic balance aliasing errors, increase the Fundamental Oversample (FundOversample) parameter on the HarmonicBalance controller from the default 1 to, say, 16.

The magnitude and phase response for Amplifier2 are fitted separately. Given, say, SOI and TOI, a set of magnitude polynomial coefficients are determined such that Amplifier2 has a second-order intercept point of SOI and a third-order intercept point of TOI. Given AM2PM and PAM2PM, a set of phase polynomial coefficients are determined such that Amplifier2 has AM to PM conversion given by AM2PM at PAM2PM. When using an Amplifier2 with both a magnitude and a phase response in a multi-tone harmonic balance simulation, there is no guarantee that the above magnitude and phase separation will hold. Generally, a phase response will generate intermodulation distortion with a multi-tone input. This is part of why AM to PM conversion is of interest. This means that the third-order intercept point may no longer be the TOI set on the component. Amplifier2 makes no attempt at matching the composite magnitude/phase third-order intercept point to the TOI set on the component but simply uses the polynomial coefficients derived for the isolated magnitude mode.

### Noise

Given the minimum noise figure NFmin (real), the optimal reflection coefficient Sopt (complex), the noise resistance Rn (real), the noise reference impedance Rref (real), and the source admittance Ys (complex), the noise figure NF of an amplifier is determined by:

$$Yopt = \frac{1}{Rref} \times \frac{1-Sopt}{1+Sopt}$$

$$Fmin = 10^{NFmin/10}$$

$$F = Fmin + \frac{Rn}{Real\,[Ys]} \times |Ys - Yopt|^2$$

$$NF = 10 \times \log(F)$$

Note that this is independent of the amplifier S-parameters.

The noise behavior of Amplifier2 is characterized by the four noise parameters NF, NFmin, Sopt and Rn and the reference impedance Z1 for port 1. Amplifier2 is implemented as a *Noisy2Port* cascaded with an SDD. The above-mentioned five parameters control the parameters for the Noisy2Port, the Noisy2Port generates a noise voltage on its output and this noise voltage is passed through the SDD in the same manner as the signal.

NF-only mode is used for NFmin=0. This is a special case where only one noise parameter must be specified. In this case, the Noisy2Port has the parameters NFmin=NF, Sopt=0, and Rn=Z1/4 × ($10^{NF/10}$-1). The reference impedance for the noise calculation (not available on the Noisy2Port user interface) is Z1. The NFmin=0, Sopt and Rn parameters are ignored.

(NFmin, Sopt, Rn) mode is used for NFmin>0. This is a more general case than NF-only mode. In this case, the Noisy2Port has the parameters NFmin=NFmin, Sopt=Sopt, and Rn=Rn. The reference impedance for the noise calculation (not available on the Noisy2Port user interface) is Z1. The NF parameter is ignored.

Given an output noise voltage *vn*, the single sideband noise figure NFssb and the double sideband noise figure NFdsb are given by:
NFssb=10*log (($vn^2$ / R + k × T0 × (G1 + G2 +...)) / (k × T0 × G1))
NFdsb=10*log( ($vn^2$ / R + k × T0 × (G1+G2 + ...)) / (k × T0 × (G1 + G2 + ...)))

where *R* is the output resistance, *k*=1.380658e-23 J/K is Boltzmann's constant, *T0*=290 K is the standard noise temperature, *G1* is the primary power gain from the input noise frequency to the output noise frequency, and *G2*+... is the sum of all higher order mixing gains which mix from some input frequency to the output noise frequency. For an amplifier, *G2*+... is zero under small-signal operation. $vn^2$ / R represents the noise added by the amplifier while *k*   *T0* represents the noise power available from the input termination. In the following, we outline how Amplifier2 calculates noise voltages and noise figures in various cases. This will be compared to the behavior of Amplifier.


**Small-Signal Operation, NF-Only Mode**
Consider an amplifier with NF=5 dB, Z1=25 Ohm and S21=1. The corresponding values of NFmin, Sopt and Rn have been described previously. We will consider the two S11 values S11=0 and S11=0.2. The amplifier lives in an environment where Ys=1/50 Siemens and R=50 Ohm. For this amplifier, the noise figure on the output should be NF=5.356 dB.

Amplifier calculates the noise voltage via:
$$Fmin = 10^{NFmin/10}$$

$$an^2 = k \times T0 \times (Fmin - 1)$$

$$bn^2 = S21^2 \times an^2$$

$$vn = \sqrt{R \times bn^2}$$

Doing the numerical evaluations, we get *vn*=657.93pV. This value is independent of S11. Because the *G*1 term used in the calculation of NFssb and NFdsb varies with S11, NFssb and NFdsb will vary with S11. Specifically, we get NFssb=NFdsb=5.356 dB for S11=0 (matches NF) and NFssb=NFdsb=4.940 dB for S11=0.2 (does not match NF).

Amplifier2 calculates the noise voltage differently. First, four noise correlation coefficients are calculated via:
< *vn* , *vn* > = 4 × *k* × *T0* × *Rn*
< *in* , *in* > = 4 × *k* × *T0* × *Rn* × |*Yopt*|$^2$
< *vn* , *in* > = 4 × *k* × *T0* × *Rn* × ((*Fmin*-1)/(2 × - ~~*Rn*~~ ) *Yopt*)
< *in* , *vn* > = conj(< *vn* , *in* >)

The noise voltage can then be calculated from:

$$vn = \sqrt{G \times \langle vn, vn \rangle \times G + G \times \langle vn, in \rangle \times Z + Z \times \langle in, vn \rangle \times G + Z \times \langle in, in \rangle \times Z}$$

where *G* is the trans-voltage-gain of the amplifier and *Z* is the trans-impedance of the amplifier. These can be found from an adjoint circuit analysis of the amplifier and become *G* =-0.4714 and *Z*=-23.5702 Ohm for S11=0 and *G*=-0.5051 and *Z*=-25.2538 Ohm for S11=0.2. The noise voltage then becomes *vn*=657.93 pV for S11=0 and *vn*=704.93 pV for S11=0.2. We see that *vn* varies with S11; this is not surprising. The noise voltage at the output of the Noisy2Port is independent of S11 but this voltage is passed through the SDD and will see a small-signal gain. NFssb and NFdsb become 5.356 dB independent of S11 (matches NF).

**Small-Signal Operation, (NFmin, Sopt, Rn) Mode**
Consider an amplifier with NFmin=5 dB, Sopt=0.1+j × 0.2, Rn=40, Z1=25 Ohm and S21=1. We will consider the two S11 values S11=0 and S11=0.2. The amplifier lives in an environment where Ys=1/(10-j × 20) Siemens and Zl=30. For this amplifier, the noise figure on the output should be NF=9.520 dB.

In this case, Amplifier2 calculates its noise voltage from the same equations as for NF-only mode. The only difference is the NFmin, Sopt and Rn values used. Amplifier2 produces the noise voltage *vn*=766.79 pV for S11=0 and *vn*=749.68 pV for S11=0.2. For both S11 values, this leads to NFssb=NFdsb=9.520 dB. Amplifier calculates noise voltages different from NF-only mode but is again wrong. Amplifier produces the noise voltage *vn*=611.47 pV for S11=0 and *vn*597.83 pV for S11=0.2. For both S11 values, this leads to NFssb=NFdsb=7.823 dB.

**Large-Signal Operation**
The above sections present the noise behavior of Amplifier and Amplifier2 in NF-only mode and (NFmin, Sopt, Rn) mode. These apply as the amplifier is operated under small-signal conditions. As the power is increased and the amplifier is operated under large-signal

conditions, the expressions for NFssb and NFdsb still apply but some of the terms change.

Consider the expressions for NFssb and NFdsb. These are:

NFssb= 10*log( ($vn^2$ / R + k × T0 × (G1 +G2 +...)) / ( k × T0 × G1))

NFdsb=10*log( ($vn^2$ / R + k × T0 × (G1 +G2 +...)) / ( k × T0 × (G1 + G2 + ...)))

At low powers, $G1$ is the amplifier's small-signal gain and $G2+...$ is zero. As the power increases and the amplifier compresses, $G1$ decreases (gain expansion could precede this trend) and $G2+...$ increases. This is a function of the signal properties of the amplifier and is independent of the amplifier's noise model. The variation of $G1$ and $G2+...$ as a function of the power level is the same for Amplifier and Amplifier2. This is not the case for the noise voltage $vn$. For Amplifier, vn is constant as the amplifier compresses. For Amplifier2, the noise voltage at the output of the *Noisy2Port* will pass through the SDD which means that $vn$ decreases as the amplifier compresses. This lowers $vn^2/R$ for Amplifier2 relative to Amplifier, causing Amplifier2 to produce smaller noise figures in compression than Amplifier.

### Notes/Equations

1.  Amplifier2 is introduced as a replacement for Amplifier. To change an existing Amplifier component to an Amplifier2 component, simply change the name from Amplifier to Amplifier2 on the schematic. The parameters for the two models are the same and Amplifier2 will adopt the values for Amplifier, making parameter re-entry unnecessary. The only exception is that the parameters GainCompType, ReferToInput and ClipDataFile will take their default values LIST, OUTPUT and yes, respectively, regardless of the values these parameters had for Amplifier. Also, the Amplifier display settings will be ignored. Amplifier2 will simply adopt its default settings, displaying S21, S11, S22 and S12.
    For examples of how to use Amplifier2, see the example
    *examples/Tutorial/Amplifier2_Example_wrk* or search the ADS examples for designs where Amplifier2 is currently used. To locate such examples, use the ADS documentation *Search* feature, search in: *Examples*, and enter *Amplifier2*. As each found item is highlighted, the ADS examples workspace directory information is displayed; from the ADS Main window select **File** > **Open** > **Example** to navigate to these examples.
    Major Differences between Amplifier and Amplifier2 summarizes the major differences between Amplifier and Amplifier2.

### Major Differences between Amplifier and Amplifier2

| Amplifier | Amplifier2 |
|---|---|
| AM to PM only supported for some magnitude modes | AM to PM supported for all magnitude modes |
| AM to PM broken | AM to PM working |
| Non-physical noise behavior | Physical noise behavior |
| Real/imaginary polynomial fit | Magnitude/phase polynomial fit |
| Complex S21 leads to non-physical time-domain waveforms | Complex S21 leads to physical time-domain waveforms |
| For large harmonic balance and circuit envelope simulations Amplifier2 may be slower than Amplifier. ||

2. Use the functions polar(mag,ang), dbpolar(dB, ang), or VSWRpolar(VSWR, ang) to convert the Sij specifications into complex numbers. Note that Sij are voltage gains and not power gains. For example, an amplifier with S21=polar(10,0) and S11=S22=S12=0 will scale the voltage by a factor of 10 from input to output and will therefore result in a 20 dB increase in power. S21=dbpolar(10,0), on the other hand, will result in a 10 dB increase in power.

3. For an S-parameter or a noise figure sinusoidal ripple, use the function ripple (mag, intercept, frequency, variable); for example ripple(0.1, 0, 10 MHz, freq).
   example: S21=dbpolar(10.0+ripple(0.1,0,10MHz,freq),0.0)

4. When defining gain using S21, remember that this gain is applied to the forward incident wave into the input of the amplifier. This is in keeping with the measurement standards used to define amplifier gain at the system level. This means that if we change S11 from 0 to 0.9 for example, we will see no change in output power because the reflected wave is not taken into account by the amplifier's definition of gain, only the incident wave.

5. Amplifier2 behaves differently from Amplifier for complex S21 values. The phase shift is applied before the nonlinear polynomial instead of after, leading to much more realistic waveforms. To see this, consider for example an Amplifier and Amplifier2 with S21=dbpolar(10,50), TOI=20 dBm, GainCompPower=10 dBm and GainComp=2 dB. Excite and terminate these with default P_1Tone and Term components and carry out a high-order harmonic balance analysis, say Order=150, at a high input power, say 30 dBm. Because the amplifiers are excited with a sine wave and operate deep into compression, we expect the output time domain waveform to be clipped and closely resemble a square wave. Except for the expected differences at transitions where the effect of a finite Order is evident, this is the case for Amplifier2 but is not at all the case for Amplifier.

6. Z1 and Z2, the reference impedance parameters for ports 1 and 2, are used in conjunction with the parameters S11/S21/S12/S22. This is because S-data is always used with respect to a particular reference impedance.

7. Amplifier2 does not support complex reference impedances.

8. Amplifier2 passes dc.

9. For circuit envelope simulation, baseband signals are blocked.

10. Amplifier2 may be slower than Amplifier for large harmonic balance and circuit envelope simulations.

11. Modeling Basics presented the polynomial model used for modeling the output voltage as a function of the input voltage. It also presented the result of pushing an $A1 \times \cos(\omega1 \times t) + A2 \times \cos(\omega2 \times t)$ signal through a $y = a1 \times x + a2 \times x^2 + a3 \times x^3$ nonlinearity. Based on this, we can calculate the IP2 (second-order intercept) and IP3 (third-order intercept) of the amplifier. The general equation for the $n$th-order intercept point IP $n$ is IP $n = (n \times P1 - Pn)/(n-1)$ where $P1$ is the power level of the first-order tone and $Pn$ is the power level of the $n$th-order tone.
    The P$n$ power level, however, is not unique. Look at the amplifier output. Ignoring $A1$

and *A*2 which we can normalize out, the second-order harmonics cos(2 × ω1 × *t*) and cos(2 × ω2 × *t*) have the amplitude 1/2 × *a*2 while the second-order intermods cos((ω1-ω2) × *t*) and cos((ω1+ω2) × *t*) have the amplitude *a*2. Similarly, the third-order harmonics cos (3 × ω1 × *t*) and cos(3 × ω2 × *t*) have the amplitude 1/4 × *a*3 while the third-order intermods cos((2 × ω1+ω2) × *t*), cos((2 × ω1-ω2) × *t*), cos((2 × 2ω+1ω) × *t*) and cos((2 × ω2-ω1) × *t*) have the amplitude 3/4 × *a*3. If the formula IP *n* = (*n* × *P*1-*P*n)/(*n*1) works for one type of second and third-order tone, it will not work for the other.

The industry standard definitions of IP2 and IP3 are based on intermods, not harmonics. When polynomial coefficients are determined in the Amplifier2 code, they are therefore based on intermods. When validating Amplifier2 using intermods, everything checks out; when using harmonics, it does not. This shows that we cannot validate the proper SOI and TOI for Amplifier2 using a one-tone analysis because this generates only harmonics but not intermods. We use industry standard definitions when deriving the coefficients for Amplifer2 and it is imperative to do the same when validating Amplifier2.

Note that SOI and TOI are defined at a low power level. If SOI and TOI are calculated at a power level where either P1 or P*n* deviate from their low-power values, the results will be in error. To see this, sweep the input power and plot the IP3 of the Amplifier2 as a function of the input power. At low powers, the IP3 of the amplifier will match the TOI parameter set on the Amplifier2. Amplifier2 was constructed to have the TOI parameter as its IP3 at low powers so this is expected. As the input power increases, IP3 will start to deviate from the TOI value for low input powers. This simply reinforces the importance of calculating IP3 at low input powers. Too high, and IP3 changes. Note that it is not enough that the fundamental tone varies linearly. SOI/TOI is calculated based on the fundamental and the second/third-order intermod so one must ensure that the second/third-order intermods are also linear or SOI/TOI will change.

Note also that Amplifier2 must be output impedance matched in order for the SOI/TOI validation to check out.

12. The common expectation for the behavior of the adjacent channel power rejection (ACPR) for an amplifier operating in compression can be formulated in many equivalent ways: the ACPR decreases/improves as TOI increases; the ACPR decreases/improves as the amplifier becomes more linear; the ACPR decreases/improves as we move away from compression; the ACPR increases/worsens as TOI decreases; the ACPR increases/worsens as the amplifier becomes more nonlinear; the ACPR increases/worsens as we move towards compression.

This expectation is correct for an amplifier operating linearly. For an amplifier operating in compression, the validity of this expectation depends on the amplifier. We will discuss two cases, a transistor-level amplifier and a polynomial amplifier. The expectation may or may not be correct for a transistor-level amplifier operating in compression. A transistor-level simulation of a particular amplifier in ADS shows that this expectation holds at some input power ranges but not at others.

The expectation is incorrect for a polynomial amplifier operating in compression. Modeling Basics presented how two tones react when pushed through a polynomial amplifier and offered a discussion of this analysis. The result is that the large-signal gain will always exceed the small-signal gain. Applied to ACPR skirts, this means that a polynomial amplifier will actually produce a decreasing/better ACPR level as we move further into compression. This may be counter-intuitive but Amplifier2 reacts the way we can theoretically predict that a polynomial amplifier must act. Amplifier2's behavior is simply a function of the polynomial modeling on which it is

based.

13. An S2D file typically consists of an ACDATA block containing small-signal information and a GCOMP *i* block (*i*=1, ... , 7) containing compression information. For Amplifier2, the ACDATA block is ignored and the S-parameters specified on the Amplifier2 component are used. If the S-parameters of the ACDATA block must be used, use the AmplifierS2D component instead. Similarly, any NDATA blocks containing noise data are ignored by Amplifier2.

14. When an S2D file contains gain compression data at more than one frequency, the GainCompFreq must be set to one of the frequencies in the S2D file to identify the data to be used. It is imperative that GainCompFreq be set to one of the frequencies in the S2D file as no interpolation or extrapolation between gain compression data at different frequencies can be performed. For further details regarding GainCompFreq selectivity, refer to *Using GCFreq to Resolve GCOMP7 Frequency Conflicts* (ccsys) in the AmplifierS2D documentation.

15. When an S2D file has a power range that exceeds that of a simulation, a choice must be made for the power range used for fitting. Assume an S2D file covers -30 dBm to 30 dBm but that a simulation is carried out from -10 dBm to 10 dBm. In this case, a choice must be made as to whether the polynomial fitting of S2D data is done over the power range -30 dBm to 30 dBm or -10 dBm to 10 dBm. In the former case, the fitting may be inaccurate as the polynomial must cover a large power range that could hold a lot of variations. This is undesirable. However, the advantage of this approach is that the results we get when simulating from -10 dBm to 10 dBm are a subset of what we would have gotten in that interval had we simulated from -30 dBm to 30 dBm. In the latter case, the fitting is much more accurate as the fitting is done over a much smaller power range which presumably holds a lot less variation. This is desirable. However, the problem with this approach is that the results between -10 dBm and 10 dBm will be different for a simulation done from -30 dBm and 30 dBm rather than from -10 dBm and 10 dBm because the polynomial coefficients change as we change the power range of the simulation. ADS does the former. It fits a polynomial to the whole S2D file, not just the subset for which the simulation is carried out. To change the fitting in a power range, it is not enough to change the power range of the simulation. To change the fitting, one must modify the S2D file. The S2D file power range, not the simulation power range, dictates the fitting power range. This is relevant in the following where we discuss different fittings in different power ranges.

16. A typical Pout (output power) vs. Pin (input power) curve consists of a linearly increasing region, a transition region and a saturation region. Another way of thinking of this is that typical Pout-Pin vs. Pin curve consists of a flat region, a transition region and a linearly decreasing region.
When the saturation region is made larger and larger, the fitting approach adopted by Amplifier2 (polynomial fitting, odd order terms, order dependent upon the number of data points in the S2D file, max order 9) will tend to produce fitted curves which ring/oscillate more and more at higher powers. Mild ringing is often tolerable and might not even be noticed but if the transition region becomes too large it can make the results useless. To alleviate the problem, reduce the size of the saturation region to the minimum needed and leave no extra points in the S2D file. If the results are still not satisfactory, ensure *ClipDataFile* is set to yes and reduce the saturation region even more, relying on Amplifier2 extrapolation. If the results are still not satisfactory, try breaking the S2D file into two files and simulate the problem in two steps. Another alternative is to use the AmpSingleCarrier model. This model is based on linear interpolation instead of curve fitting and will not have this ringing problem. AmpSingleCarrier, however, will not produce harmonics. Refer to the

AmpSingleCarrier documentation for details.

If fitted results do not accurately match the data in the S2D file and it is uncertain if this ringing problem is the cause, the problem is very easy to exaggerate. Simply extend the GCOMP7 block of the S2D file with a large flat region (more input powers, saturated output power, saturated output phase) and re-simulate. If the ringing problem is the cause, the results should get worse.

17. The S2D file capability is a legacy from OmniSys. OmniSys used GComp1-GComp7 data items for specifying gain compression. *Gain Compression Data for OmniSys and ADS* summarizes the gain compression data for OmniSys and ADS. Refer to OmniSys Parameter Information for OmniSys parameter information. GComp1-GComp6 can be specified by using the corresponding ADS gain compression parameters and setting GainCompType=LIST or they can be contained in an S2D format setting GainCompType=FILE.

| OmniSys | ADS |
|---|---|
| GComp1: IP3 | TOI |
| GComp2: 1dBc | GainComp=1dB |
| GComp3: IP3, 1dBc | TOI |
| GComp4: IP3, Ps, GCS | TOI |
| GComp5: 1dBc, Ps, GCS | GainComp=1dB |
| GComp6: IP3, 1dBc, Ps, GCS | TOI |
| GComp7 | GainCompType=FILE |



**OmniSys Parameter Information**

18. The GainRF component is the Ptolemy equivalent of the Analog/RF Amplifier2 component. Both of these components are based on the OmniSys legacy (refer to note 16) and the OmniSys cases GCOMP1 through GCOMP7 are shared by GainRF and Amplifier2. For GCOMP1 through GCOMP6 corresponding to GainCompType=LIST, the curve fitting algorithms for GainRF and Amplifier2 are very similar and close results can be expected. However, the curve fitting algorithms are not identical and the shape of the knees of the compression curves will therefore differ slightly. Also, the levels at which the various fundamentals saturate can be different. These levels will generally differ more when Psat is not set than when Psat is set. For GCOMP7 corresponding to GainCompType=FILE, the curve fitting

algorithms for GainRF and Amplifier2 are different and different results can be expected.

# Amplifier (Obsolete RF System Amplifier)

## Symbol
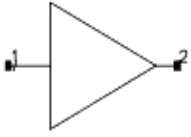


> **ⓘ Note**
> 2003C introduces an improved version of Amplifier: Amplifier2. Please use Amplifier2 for new designs; refer to Amplifier2 documentation for more information.

## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| S21 | Forward transmission gain (real or complex number; refer to note 2) | None | None |
| S11 | Port 1 reflection (real or complex number; refer to note 2) | None | None |
| S22 | Port 2 reflection (real or complex number; refer to note 2) | None | None |
| S12 | Reverse transmission gain (real or complex number; refer to note 2) | None | None |
| NF | Input noise figure | dB | None |
| NFmin | Minimum noise figure at Sopt | dB | None |
| Sopt | Optimum source reflection for NFmin | None | None |
| Rn | Equivalent noise resistance | None | None |
| Z1 | Reference impedance for port 1 (real or complex number) | None | None |
| Z2 | Reference impedance for port 2 (real or complex number) | None | None |
| ClipDataFile | Clip data beyond maximum input power: NO to disable; YES to enable | None | NO |
| ImpNoncausalLength | Non-causal function impulse response order | Integer | None |
| ImpMode | Convolution mode | Integer | None |
| ImpMaxFreq | Maximum frequency to which device is evaluated | None | None |
| ImpDeltaFreq | Sample spacing in frequency | None | None |
| ImpMaxOrder | Maximum allowed impulse response order | Integer | None |
| ImpWindow | Smoothing window | Integer | None |
| ImpRelTol | Relative impulse response truncation factor | None | None |
| ImpAbsTol | Absolute impulse response truncation factor | None | None |

## Range of Usage

NF ≥ 0 dB
NFmin > 0
0 < | Sopt | < 1
0 < Rn

GainCompFreq > 0
For S21 = mag/ang
$|S_{21}| > 0$

## Gain Compression Parameters

| Name | Description | Units | Default |
|---|---|---|---|
| GainCompType | Gain compression type: | None | None |
| GainCompFreq | Reference frequency for gain compression (if gain compression is described as a function of frequency) | None | None |
| ReferToInput | Specify gain compression with respect to input or output power of device | None | None |
| SOI | Second order intercept | dBm | None |
| TOI | Third order intercept | dBm | None |
| Psat | Power level at saturation | dBm | None |
| GainCompSat | Gain compression at Psat | dB | None |
| GainCompPower | Power level in dBm at gain compression for X dB compression point, specified by GainComp | dBm | None |
| GainComp | Gain compression at GainCompPower | dB | 1 |
| AM2PM | Amplitude modulation to phase modulation | deg/dB | None |
| PAM2PM | Power level at AM2PM | dBm | None |
| GainCompFile | Filename for gain compression data in S2D file format | None | None |

## Range of Usage for Gain Compression Parameters

When specifying gain compression using model parameters, only certain combination of parameters will produce stable polynomial curve fitting. The recommended parameter combinations are listed here.

> **Note** If unrealistic parameter values are used, the polynomial will become unstable, resulting in oscillations.

- Third-order intercept and 1dB gain compression parameters:
  TOI, GainCompPower with GainComp=1dB
  Range of validity: TOI > GainCompPower + 10.8
- Third-order intercept and power saturation parameters:
  TOI, Psat, GainCompSat
  Range of validity: TOI > Psat + 8.6
- 1dB gain compression and power saturation parameters:
  GainCompPower with GainComp=1dB, Psat, GainCompSat
  Range of validity: Psat > GainCompPower + 3
- Third-order intercept, 1dB gain compression and power saturation parameters:
  TOI, GainCompPower with GainComp=1dB, Psat, GainCompSat
  Range of validity: Psat > GainCompPower +3, TOI > GainCompPower + 10.8
- Second-order intercept and third-order intercept parameters: SOI, TOI
- AM to PM with 1dB gain compression parameters:
  AM2PM, PAM2PM, and GainCompPower with GainComp=1dB
  The value for AM2PM must satisfy this condition to avoid a square root of a negative number:

$$AM2PM < \frac{180}{\pi} \times 10^{((PAM2PM - GainCompPower)/10)} \times 10^{(-3 \times (GainComp)/20)}$$

$$\frac{1}{2.34}$$

- AM to PM with third-order intercept parameters:
  AM2PM, PAM2PM, and TOI
  The value for AM2PM must satisfy this condition to avoid a square root of a negative number:

$$AM2PM < \frac{\left(\frac{180}{\pi}\right) \times 10^{((PAM2PM - TOI)/10)}}{2.34}$$

  If SOI is not specified, the amplifier is modeled using a polynomial of odd orders:

  y = a1 × x + a3 × x$^3$ + a5 × x$^5$ + . . . .
  As a result, only odd order harmonics (m × f, where m is an odd number) and odd order intermods (m × f1 + n × f2, where m+n is an odd number) are taken into account.
  If SOI is specified, the amplifier polynomial has an even order term:

  y = a1 × x + a2 × x$^2$ + a3 × x$^3$ + a5 × x$^5$ + . . . .
  As a result, both odd and even order harmonics and intermods are taken into account in the simulation.

**Warning Messages**

When values for TOI, 1 GainCompPower, and Psat are properly related, the DC input-output transfer characteristic has the form shown in DC Input-Output Transfer Characteristics.

- No Saturation. A warning is displayed if a polynomial is generated that does not have a maximum where the transfer characteristics can be clipped (when the amplifier cannot reach saturation). Refer to No Saturation Warning.

- Non-Monotonic Transfer Curve. A warning is displayed if the value specified for Psat is lower than GainCompPower. Refer to Non-Monotonic Transfer Curve Warning.

The result of this specification is that the saturated output is lower than the output at the 1 dB compression point, and the input-output characteristics have a non-monotonic characteristic transfer curve.

**Figure: DC Input–Output Transfer Characteristics**
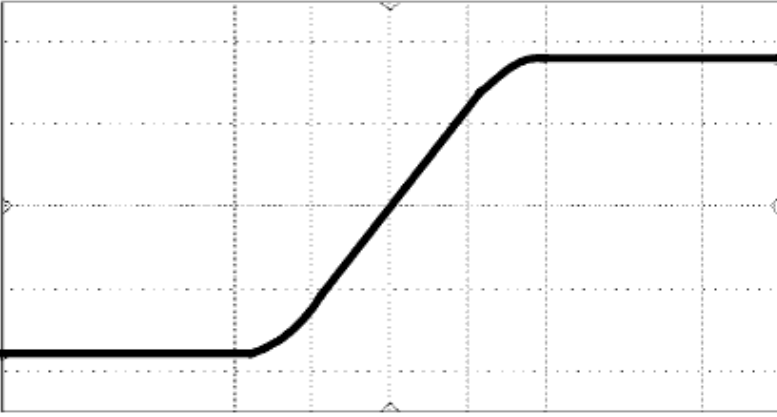
**Figure: No Saturation Warning**
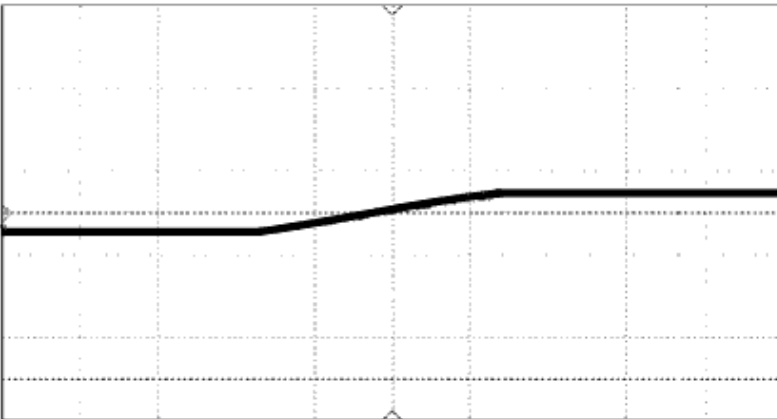


**Figure: Non-Monotonic Transfer Curve Warning**
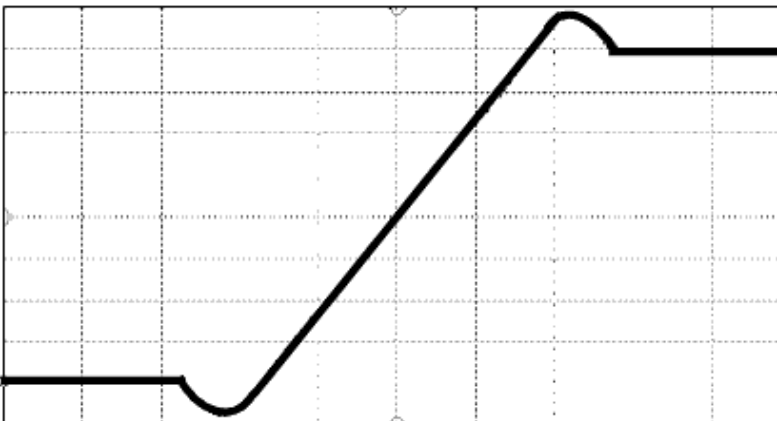


**Notes/Equations**

1. If NFmin, Sopt, and Rn are used to characterize noise, the following relation must be satisfied for a realistic model.

$$\frac{Rn}{Zo} \geq \frac{To(Fmin-1)|1+Sopt|^2}{T\,4} \frac{(1-|S_{11}|^2)}{|1-Sopt\,S_{11}|^2}$$

   A warning message will be issued if Rn does not meet this criterion. If the noise parameters attempt to describe a system that requires negative noise (due to Rn being too small), the negative part of the noise will be set to zero and a warning message will be issued.

2. Use the function polar(mag,ang), or dbpolar(dB, ang), or VSWRpolar(VSWR, ang) to convert these specifications into a complex number.

3. For an S-parameter or a noise figure sinusoidal ripple, use the function ripple (mag, intercept, period, variable); for example ripple(0.1, 0, 10 MHz, freq).
   example: S21=dbpolar(10+ripple( ),0.)

4. When you define the gain using S21, keep in mind that this gain is applied to the forward incident wave into the input of the amplifier. This is in keeping with the measurement standards used to define amplifier gain at a system level. This means that if you change S11 from 0 to 0.9 for example, you will see no change in output power because the reflect wave is not taken into account by the amplifier's definition of gain, only the incident wave.

5. Z1 and Z2, the reference impedance parameters for ports 1 and 2, are used in conjunction with the parameters S11/S21/S12/S22. This is because S-data is always used with respect to a particular reference impedance.

6. This model blocks dc.

7. For circuit envelope simulation, baseband signals are blocked.

8. OmniSys used GComp1-GComp7 data items for specifying gain compression. *Gain Compression Data for OmniSys and ADS* summarizes the gain compression data for OmniSys and ADS. Refer to OmniSys Parameter Information for OmniSys parameter information.
   GComp1-GComp6 can be specified by using the corresponding ADS gain compression parameters and setting GainCompType=LIST. Or, they can also be contained in an S2D format setting GainCompType=FILE. Also note that an S2D file could contain other data such as small signal S-parameters and noise; these data are ignored by the RF System Amplifier.

| OmniSys | ADS |
|---------|-----|
| GComp1: IP3 | TOI |
| GComp2: 1dBc | GainComp=1dB<br>GainCompPower |
| GComp3: IP3, 1dBc | TOI<br>GainComp=1dB<br>GainCompPower |
| GComp4: IP3, Ps, GCS | TOI<br>Psat<br>GainCompSat |
| GComp5: 1dBc, Ps, GCS | GainComp=1dB<br>GainCompPower<br>Psat<br>GainCompSat |
| GComp6: IP3, 1dBc, Ps, GCS | TOI<br>GainComp=1dB<br>GainCompPower<br>Psat<br>GainCompSat |
| GComp7 | GainCompType=FILE<br>GainCompFile=filename |

**Figure: OmniSys Parameter Information**
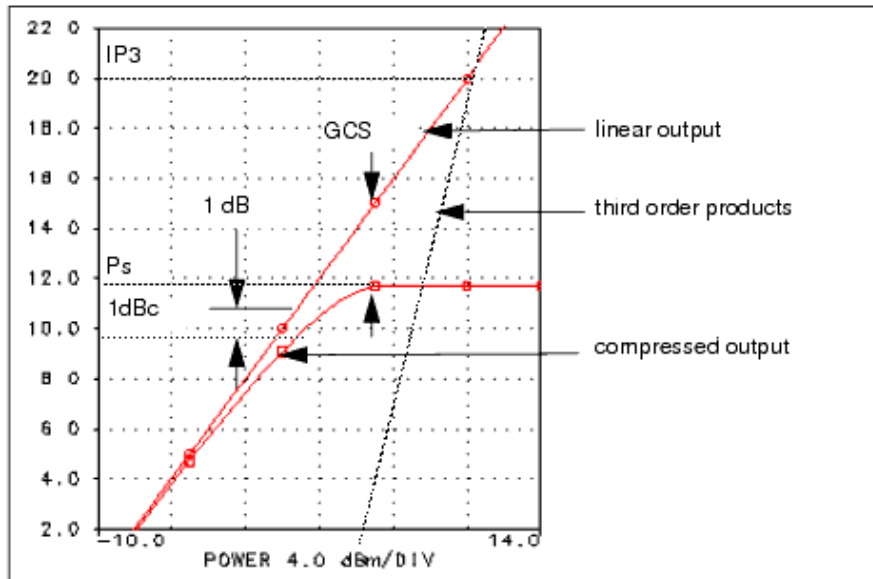


9. The AM to PM option uses parabolic amplitude dependence to describe the amplitude to phase modulation conversion. When a signal of type $V_{in}(\tau)=A\cos(W\varphi)$, is applied to the input of a device with parabolic AM to PM, the output phase exhibits:

$$Phase(V_{out}(\tau)) = kA^2$$

Therefore, this phase depends on the input signal amplitude A in a parabolic manner. Because the conversion of amplitude to phase is amplitude dependent, the AM to PM (AM2PM) is specified in degrees per decibel at a given output power (PAM2PM). k is calculated from these two parameters.

$$k = \frac{AM2PM\left(\frac{\pi}{180}\right)2.34}{10^{\left(\frac{PAM2PM-30}{10}\right)}}$$

When AM to PM is specified, the third-order intermod and gain compression are side effects. If AM2PM is specified to be very large compared to the third-order intercept or gain compression, a warning is issued.

# AmplifierVC (Ideal Voltage-Controlled Amplifier)

## Symbol
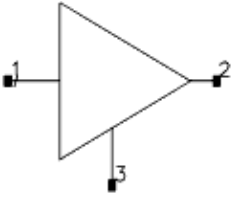


## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Gain | Gain as a function of control voltage (_v3) | dB/V | (30 − 15 × _v3) |
| Rout | Output resistance | Ohm | 50 |

## Range of Usage

Rout > 0

## Notes/Equations

1. AmplifierVC is an ideal voltage controlled amplifier. The impedance for its input and control port is infinite; its output impedance is Rout.
2. An equation is used to describe Gain as a function of the control voltage at port 3: _V3. The default equation is Gain = 30 − 15 × _V3.
3. The AmplifierVC response cannot depend upon frequency.

# AmpSingleCarrier (Single Carrier Amplifier)

## Symbol
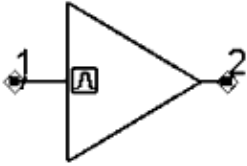


## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Freq | Frequency | None | None |
| S21 | Forward transmission coefficient; use x + j × y polar (x,y), dbpolar (x, y) for complex value | None | None |
| S11 | Forward reflection coefficient; use x + j × y polar (x,y), dbpolar (x, y), vswrpolar (x, y) for complex value | None | None |
| S22 | Reverse reflection coefficient; use x + j × y polar (x,y), dbpolar (x, y), vswrpolar (x, y) for complex value | None | None |
| S12 | Reverse reflection coefficient; use x + j × y polar (x,y), dbpolar (x, y) for complex value | None | None |
| NF | Noise figure | dB | None |
| NFmin | Minimum noise figure at Sopt | dB | None |
| Sopt | Optimum source reflection for minimum noise figure; use x + j × y polar (x,y), dbpolar (x, y) for complex value | None | None |
| Rn | Equivalent noise resistance | None | None |
| Z1 | Reference impedance for port 1 | None | None |
| Z2 | Reference impedance for port 2 | None | None |
| GainCompFile | Filename for Gain Compression (GCOMP7) Data | None | None |

## Notes/Equations

1. AmpSingleCarrier is based on FDD (frequency-domain defined device). The incident power level at the amplifier input is detected and the corresponding gain is obtained by interpolating the gain compression table given by GainCompFile. In harmonic balance simulation, the output signal has only one frequency component given by the Freq parameter. Neither harmonics nor intermodulation products are generated by AmpSingleCarrier. In Envelope simulation, the output signal contains only the complex envelope around one frequency given by the Freq parameter.

2. Z1 and Z2, the reference impedance parameters for ports 1 and 2, are used in conjunction with the parameters S11/S21/S12/S22. This is because S-data is always used with respect to a particular reference impedance.

# FreqMult (Ideal Frequency Multiplier)

## Symbol
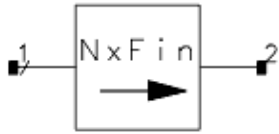


## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| S11 | Complex reflection coefficient for port 1 | None | 0 |
| S22 | Complex reflection coefficient for port 2 | None | 0 |
| G1 | Power gain of input tone | dB | 3 |
| G2 | Power gain of second harmonic relative to input tone | dB | None |
| G3 | Power gain of third harmonic relative to input tone | dB | None |
| G4 | Power gain of fourth harmonic relative to input tone | dB | None |
| G5 | Power gain of fifth harmonic relative to input tone | dB | None |
| G6 | Power gain of sixth harmonic relative to input tone | dB | None |
| G7 | Power gain of seventh harmonic relative to input tone | dB | None |
| G8 | Power gain of eighth harmonic relative to input tone | dB | None |
| G9 | Power gain of ninth harmonic relative to input tone | dB | None |
| Pmin | Minimum input power for specified conversion | dBm | −40 |
| Z1 | Reference impedance for port 1 | Ohm | 50 |
| Z2 | Reference impedance for port 2 | Ohm | 50 |

## Range of Usage

$0 \leq S11 < 1$
$0 \leq S22 < 1$

## Notes/Equations

1. The ideal frequency multiplier takes an input signal and produces an output spectrum with specified spectral harmonics. The reverse isolation is assumed to be infinite (S12=0). All of the harmonics generation is specified relative to the input level. For example if an input power of 20 dBm is incident on a multiplier with G2=−20 dB the second harmonic output will be 0 dBm. This device is compatible with transient simulation.

2. This model assumes that only one signal tone is present at the input. If multiple tones are used at the input then unwanted mixing products can be generated and spurious mixing products will result.

3. The harmonic balance parameter ORDER must be set to a value equal to or higher than the harmonic index of interest.

4. Real-world nonlinear devices such as mixers and frequency multipliers often have an input power level below which they do not work. For FreqMult, this phenomenon is incorporated through the PMin parameter. However, PMin is not simply a minimum threshold value to which the input power is limited.

   With a right-propagating input wave of a1, the input power detection is done via $|a1|^2$ =a1 × a1+H(a1) × H(a1), with H(a1) being the Hilbert transform of a1. A normalized a1 is then calculated via a1norm=a1/sqrt($|a1|^2$+dBm2lin(PMin)), with dBm2lin(x)=$10^{(x-30)/10}$. Note the presence of PMin. For dBm2lin(PMin)<<$|a1|^2$, the effect of PMin is negligible. For $|a1|^2$ approaching dBm2lin(PMin), however, the results will depend on the value of PMin. If this is undesired, simply lower PMin appropriately.

5. Z1 and Z2, the reference impedance parameters for ports 1 and 2, are used in conjunction with the parameters S11/S21/S12/S22. This is because S-data is always used with respect to a particular reference impedance.

# LogACDemod (Demodulating AC Logarithmic Amplifier)

**Symbol**
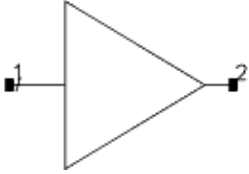


**Parameters**

| Name | Description | Units | Default |
|---|---|---|---|
| CurrentSlope | Gradient of transfer characteristic | A/decade | 1.e-3 |
| VoltIntercept | Vin for zero output | V | 1.e-3 |
| Z1 | Reference impedance for port 1 | Ohm | 50 |

**Notes/Equations**

1. LogACDemod uses a square-law detector and an ideal logarithmic function.
2. The function is in the form of voltage-input/current-output.
3. LogACDemod is not recommended for transient simulation.
4. Several logarithmic amplifiers are available in ADS: LogACDemod, LogDC, LogSuccDetect, and LogTrue. These components accept an RF input and generate a DC output voltage proportional to the log of the input voltage.

# LogDC (DC Logarithmic Amplifier)

## Symbol
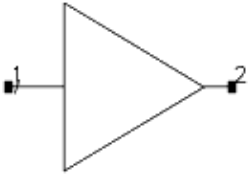
## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| VoltSlope | Gradient of transfer characteristic | V/decade | 1.e-3 |
| VoltIntercept | Vin for zero output | V | 1.e-3 |
| Z1 | Reference impedance for port 1 | Ohm | 50 |

## Notes/Equations

1. LogDC provides an output that is a logarithmic function of the input. If, for example, the scaling is 1 volt/decade, the output changes by 1V for any tenfold increase of the input.
   The intercept point is the input level at which the output voltage is 0.
   Vout = VoltSlope × log ((Vin / VoltIntercept) + 1)
2. Several logarithmic amplifiers are available in ADS: LogACDemod, LogDC, LogSuccDetect, and LogTrue. LogACDemod, LogSuccDetect, and LogTrue accept an RF input and generate a DC output voltage proportional to the log of the input voltage. LogDC will accept an RF input but does not generate a DC output.

# LogSuccDetect (Successive Detection Logarithmic Amplifier)
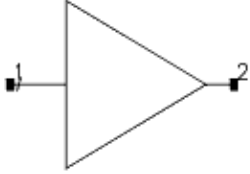
## Symbol



## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| NumStages | Number of stages | None | 5 |
| StageGain | Gain per stage | dB | 10 |
| CurrentSlope | Gradient of transfer characteristic | A/decade | 1.e-3 |
| Z1 | Reference impedance for port 1 | Ohm | 50 |

## Notes/Equations

1. This amplifier use a successive detection scheme to provide an output current proportional to the logarithm of the input voltage. The amplifier consists of several amplifier/limiter stages (NumStages specifies how many), each having a small signal gain (StageGain specifies the gain). Each stage has an associated full-wave detector, whose output current depends on the absolute value of its input stage. These output currents are summed to provide the output scaled at the CurrentSlope (amp/decade). The output contains amplitude information only, regardless of any phase information. A simplified block diagram of this component is shown in Simplified Block Diagram of a 5-Stage LogSuccDetect.

2. Several logarithmic amplifiers are available in ADS: LogACDemod, LogDC, LogSuccDetect, and LogTrue. These components accept an RF input and generate a DC output voltage proportional to the log of the input voltage.

# LogTrue (True Logarithmic Amplifier)
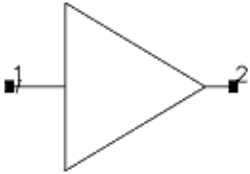
## Symbol



## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| NumStages | Number of stages | None | 5 |
| StageGain | Gain per stage | dB | 10 |
| VoltLimit | Limiting voltage of each stage | V | 1 |
| Z1 | Reference impedance for port 1 | Ohm | 50 |

## Notes/Equations

1. LogTrue accepts inputs of either polarity and generates an output whose sign follows that of the input. A progressive compression technique is used, in which the logarithmic response can be approximated through the use of cascaded amplifier stages that have signal-dependent gain.
2. Several logarithmic amplifiers are available in ADS: LogACDemod, LogDC, LogSuccDetect, and LogTrue. These components accept an RF input and generate a DC output voltage proportional to the log of the input voltage.

# Mixer2 (RF System Mixer)

## Symbol
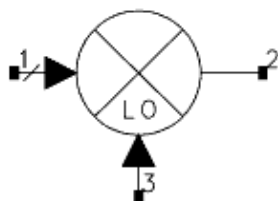


## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| SideBand | Specify the sideband/image option for the mixer (See note 1) | None | BOTH |
| OutputSidebandSuppression | Output sideband suppression (only relevant for SideBand=LOWER, UPPER) | dB | −200 |
| InputImageRejection | Input image rejection (only relevant for SideBand=LOWER IMAGE REJECTION, UPPER IMAGE REJECTION) | dB | −200 |
| ConvGain | Conversion gain; use x + j × y, polar(x,y), dbpolar(x,y) for complex value | None | dbpolar(0,0) |
| RevConvGain | Reverse conversion gain, use x + j × y, polar(x,y), dbpolar(x,y) for complex value | None | polar(0,0) |
| SP11 | S11, RF port reflection, use x + j × y, polar(x,y), dbpolar(x,y), vswrpolar(x,y) for complex value | None | polar(0,0) |
| SP12 | S12, IF port to RF port leakage, use x + j × y, polar(x,y), dbpolar(x,y), vswrpolar(x,y) for complex value | None | polar(0,0) |
| SP13 | S13, LO port to RF port leakage, use x + j × y, polar(x,y), dbpolar(x,y), vswrpolar(x,y) for complex value | None | polar(0,0) |
| SP21 | S21, RF port to IF port leakage, use x + j × y, polar(x,y), dbpolar(x,y), vswrpolar(x,y) for complex value | None | polar(0,0) |
| SP22 | S22, IF port reflection, use x + j × y, polar(x,y), dbpolar(x,y), vswrpolar(x,y) for complex value | None | polar(0,0) |
| SP23 | S23, LO port to IF port leakage, use x + j × y, polar(x,y), dbpolar(x,y), vswrpolar(x,y) for complex value | None | polar(0,0) |
| SP31 | RF port to LO port leakage (real or complex number)S31, RF port to LO port leakage, use x + j × y, polar(x,y), dbpolar(x,y), vswrpolar(x,y) for complex value | None | polar(0,0) |
| SP32 | IF port to LO port leakage (real S32, IF port to LO port leakage, use x + j × y, polar(x,y), dbpolar(x,y), vswrpolar(x,y) for complex value or complex number) | None | polar(0,0) |
| SP33 | LO port reflection (real or complex numS33, LO port reflection, use x + j × y, polar(x,y), dbpolar(x,y), vswrpolar(x,y) for complex valueor number) | None | polar(0,0) |
| PminLO | Minimum LO power before starvation | dBm | −100 |
| DetBW | Detector bandwidth for LO limiting | Hz | 1e100 |
| NF | Double sideband noise figure | dB | None |

| | | | |
|---|---|---|---|
| NFmin | Minimum double sideband noise figure at Sopt | dB | None |
| Sopt | Optimum Source Reflection for Minimum Noise Figure, use x + j × y, polar(x,y), dbpolar(x,y) for complex value | None | None |
| Rn | Equivalent noise resistance | | |
| Z1 | Reference impedance for port 1 (must be a real number) | **50** | |
| Z2 | Reference impedance for port 2 (must be a real number) | **50** | |
| Z3 | Reference impedance for port 3 (must be a real number) | **50** | |
| GainCompType | Gain compression type: | None | LIST |
| GainCompFreq | reference frequency for gain compression if gain compression is described as a function of frequency | | |
| ReferToInput | Specify gain compression with respect to input or output power of device: | None | OUTPUT |
| SOI | Second order intercept | dBm | None |
| TOI | Third order intercept | dBm | None |
| Psat | Power saturation point (always referred to output, regardless of the value of the ReferToInput parameter) | dBm | None |
| GainCompSat | Gain compression at Psat | dB | 5.0 |
| GainCompPower | Power level in dBm at gain compression for X dB compression point, specified by GainComp | dBm | None |
| GainComp | Gain compression at GainCompPower | dB | 1.0 dB |
| AM2PM | Amplitude modulation to phase modulation | deg/dB | None |
| PAM2PM | Power level at AM2PM | dBm | None |
| GainCompFile | Filename for gain compression data in S2D file format | None | None |
| ClipDataFile | Clip data beyond maximum input power: YES=enable, NO=disable | None | YES |
| ImpNoncausalLength | Non-causal function impulse response order | Integer | None |
| ImpMode | Convolution mode | Integer | None |
| ImpMaxFreq | Maximum frequency to which device is evaluated | GHz | 10 |
| ImpDeltaFreq | Sample spacing in frequency | | |
| ImpMaxOrder | Maximum allowed impulse response order | Integer | 32 |
| ImpWindow | Smoothing window | Integer | 0 |
| ImpRelTol | Relative impulse response truncation factor | None | None |
| ImpAbsTol | Absolute impulse response truncation factor | None | None |

**Frequently Asked Questions**

**Q1** : What are the major differences between Mixer and Mixer2?
**A1** : Refer to *note 1* (ccsys).

**Q2** : What are the supported parameter combinations?
**A2** : Refer to Range of Usage .

**Q3** : What is the range of usage for each parameter combination?
**A3** : Refer to Range of Usage.

**Q4** : What are the port numbers and the terminology used for the signals at each port?
**A4** : Refer to Terminology.

**Q5** : What is the basic implementation of Mixer2?
**A5** : A *Noisy2Port* cascaded with an *SDD* . Refer to <u>Basic Implementation</u> for details.

**Q6** : How does Mixer2 operate for the various SideBand modes?
**A6** : Refer to <u>Sideband Suppression and Image Rejection</u>.

**Q7** : What is ConvGain?
**A7** : It is the conversion gain. Refer to <u>Linear Behavior</u> for details.

**Q8** : How does Mixer2 apply a complex ConvGain?
**A8** : Before the mixing, not after. Refer to <u>Linear Behavior</u> for details.

**Q9** : What is RevConvGain?
**A9** : It is the reverse conversion gain. Refer to <u>Linear Behavior</u> for details.

**Q10** : What are SPij (i,j=1,2,3)?
**A10** : They are the reflection/leakage/isolation parameters or S-parameters. Refer to <u>Linear Behavior</u> for details.

**Q11** : Are ConvGain, RevConvGain and SPij (i,j=1,2,3) wave-based or voltage-based?
**A11** : ConvGain is voltage-based, RevConvGain and SPij are wave-based. Refer to <u>Linear Behavior</u> for details.

**Q12** : How is RF to IF compression modeled?
**A12** : A polynomial compression model is used. Refer to <u>Nonlinear Behavior</u> and *Modeling Basics* (ccsys) of the Amplifier2 documentation for details.

**Q13** : How is AM to PM conversion modeled?
**A13** : A polynomial AM to PM conversion model is used. Refer to *AM to PM Conversion* (ccsys) of the Amplifier2 documentation for details.

**Q14** : How is LO limiting done for different DetBW values?
**A14** : Via a filter with a variable bandwidth specified by DetBW or via a Hilbert transform. Refer to <u>LO Limiting</u> for details.

**Q15** : What's best, small or large DetBW values?
**A15** : Small for LO signals with significant harmonic content, large for bandpass LO signals. Refer to <u>LO Limiting</u> for details.

**Q16** : How does PminLO influence the LO limiting?
**A16** : It defines a lower limit for the effect of the LO power on the mixer's conversion gain. Refer to <u>LO Limiting</u> for details.

**Q17** : Why does the mixer's conversion gain vary depending on the frequency content of the LO signal?
**A17** : LO limiting is dynamic. Refer to <u>LO Limiting</u> for details.

**Q18** : What noise model is used by Mixer2?
**A18** : A *Noisy2port* , similar to Amplifier2. Refer to <u>Noise</u> for details.

**Q19** : What are "NF-only mode" and "(NFmin,Sopt,Rn) mode" for in Mixer2?
**A19** : They are two different ways of specifying noise. Refer to <u>Noise</u> for details.

**Q20** : How is NFssb/NFdsb calculated, both at low powers and in compression?
**A20** : Refer to [Noise](Noise).

**Q21** : Can you provide more details about the noise voltages and noise figures produced by Mixer2 for the different SideBand modes of operation?
**A21** : Refer to [Noise](Noise).

**Q22** : How can I lower the noise at the output of Mixer2?
**A22:** Refer to [Noise](Noise).

**Q23** : Is there a tutorial example for Mixer2?
**A23:** *examples/Tutorial/Mixer2_Example_wrk* . Refer to *note 2* for details.

**Q24** : Does Mixer2 support frequency conversion AC (FCAC) analysis?
**A24** : No. Refer to *note 3* for details.

**Q25** : Why doesn't Mixer2 work for complex reference impedances?
**A25** : Mixer2 does not support complex reference impedances. Refer to *note 4* for details.

**Q26** : Can I use Mixer2 for baseband envelope simulations?
**A26** : No. Refer to *note 6* for details.

**Q27** : Why is Mixer2 sometimes slower than Mixer?
**A27** : This is a consequence of the implementational differences between Mixer and Mixer2. Refer to *note 7* for details.

**Q28** : Why don't the predicted second- and third-order intercept points match SOI and TOI as set on Mixer2?
**A28** : You are probably not setting up your validation correctly. Make sure to do a two-tone simulation, not a one-tone simulation. Refer to *note 10* for details.

**Q29** : Do sums/differences of two RF frequencies appear at the output of Mixer2?
**A29** : It depends on SP21 and SOI. Refer to *note 11* for details.

**Q30** : Why does Mixer2 ignore the ACDATA block of my S2D file?
**A30** : It uses the S-parameters on the component instead. Refer to note 12 for details.
**Q31** : When the power range in an S2D file differs from that of a simulation, which power range is used for polynomial fitting?
**A31** : The power range in the S2D file. Refer to *note 14* for details.

**Q32** : Why do I get ringing at high powers when using an S2D file?
**A32** : This is a consequence of the polynomial model adopted by Mixer2. Refer to *note 15* for details.

**Q33** : How do I get rid of this ringing?
**A33:** Eliminate data for high input powers, rely on extrapolation via ClipDataFile=yes, or break the S2D file into two. Refer to *note 15* for details.

**Range of Usage**

NF ≥ 0 dB
NFmin ≥ 0 dB
0 < Sopt < 1
Rn > 0
GainCompFreq > 0
0 dB < GainComp < 3 dB
When specifying gain compression using model parameters, only certain combination of parameters will produce stable polynomial curve fitting. If unrealistic parameter values are used, the polynomial will become unstable, resulting in oscillations. The recommended parameter combinations are listed here:

- Third-order intercept parameter:
  Parameters: TOI
  Range of validity: N/A
- Gain compression parameters:
  Parameters: GainCompPower, GainComp
  Range of validity: N/A
- Power saturation parameters:
  Parameters: Psat, GainCompSat
  Range of validity: N/A
- Third-order intercept and 1dB gain compression parameters:
  Parameters: TOI, GainCompPower with GainComp=1dB
  Range of validity: TOI > GainCompPower + 10.8
- Third-order intercept and power saturation parameters:
  Parameters: TOI, Psat, GainCompSat
  Range of validity: TOI > Psat + 8.6
- 1dB gain compression and power saturation parameters:
  Parameters: GainCompPower with GainComp=1dB, Psat, GainCompSat
  Range of validity: Psat > GainCompPower + 3
- Third-order intercept, 1dB gain compression and power saturation parameters:
  Parameters: TOI, GainCompPower with GainComp=1dB, Psat, GainCompSat
  Range of validity: Psat > GainCompPower +3, TOI > GainCompPower + 10.8
- Second-order intercept and third-order intercept parameters:
  Parameters: SOI, TOI
  Range of validity: N/A

## Terminology

This section outlines the terminology that will be used in the remainder of the Mixer2 documentation.
Mixer2 is a 3-port device. Its input port is numbered port 1, its output port is numbered port 2, and its local oscillator port is numbered port 3. The signals at the three ports will be denoted by RF, IF, and LO, respectively, as illustrated in Mixer Terminology.. This follows the receiver/down-conversion convention. This convention has RF and IF flipped relative to the transmitter/up-conversion convention. When Mixer2 is used for transmitter/up-conversion applications, IF therefore refers to the input signal and RF refers to the output signal.

Input                          Output

Image        1 —— RF ⊗ IF ——2    Sideband

Rejection                      Suppression

                    LO

                    3
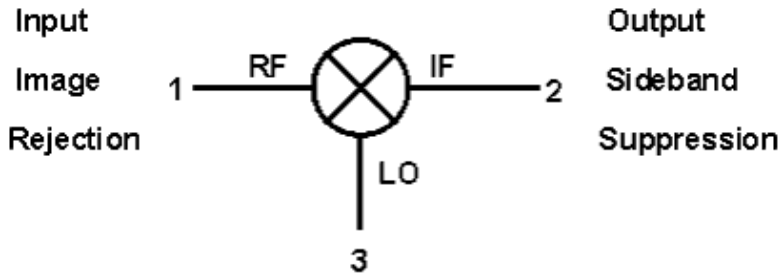
## Mixer Terminology.

The primary signal of interest at the *input* of the mixer is the RF signal. A secondary signal of interest at the input of the mixer is the RF image, denoted RFimg, defined as the other input signal which results in IF as a second-order mixing product. The input image is typically only of interest for receiver/down-conversion applications, in which case it is given by RFimg=2*LO-RF. RF and RFimg=2*LO-RF both mix down with LO to generate IF=|LO-RF| at the output. The signals RF and RFimg are called *images* of each other with respect to LO. If RFimg is attenuated relative to RF, we talk about image *rejection* .
The primary signals of interest at the *output* of the mixer are |LO-RF| and LO+RF. Typically, one of these is desired and the other is undesired. We refer to these two signals as lower and upper *sidebands* . If one sideband is attenuated relative to the other, we talk about sideband *suppression* .
Throughout the Mixer2 documentation, we will use the terms image and rejection when describing input signals to Mixer2 and use the terms sideband and suppression when describing output signals from Mixer2. Other choices are possible but the terminology adopted here is in line with accepted industry literature, see for example the exhaustive mixer applications notes from WJ Communications. The only exception is that the Mixer2 SideBand parameter controls both the image and the sideband operation of the mixer, not just the sideband operation. The reason for this is compatibility with previous versions of Mixer2.

## Basic Implementation

The basic implementation of Mixer2 is a Noisy2Port cascaded with an SDD. The parameters NF, NFmin, Sopt, Rn and Z1 are passed to the Noisy2Port and noise is generated. The signal is not affected by the generation of noise but goes unaltered through the Noisy2Port. All parameters except NF, NFmin, Sopt and Rn are then passed to the SDD and the signal and noise goes through the SDD. The SDD implements a linear mixer which allows compression from RF to IF and from RF to LO. Different circuit topologies are used for realizing sideband suppression and image rejection mixers, but the overall implementation is the same - a Noisy2Port cascaded with an SDD. This basic knowledge of the implementation may help understand some of the following sections.

## Sideband Suppression and Image Rejection

A mixer can pass both the lower and the upper input image, reject the lower input image, or reject the upper input image. Similarly, a mixer can generate both the lower and the

upper output sideband, generate the lower sideband only, or generate the upper sideband only. Various combinations of these are possible. To explain the image rejection and sideband suppression features for Mixer2, we will look at a receiver/down-conversion example.

Given RF and LO for a down-converting mixer, we have IF=|LO-RF|. Also of interest is the RF image given by RFimg=2*LO-RF. If RF is below LO, RFimg is above LO. Similarly, if RF is above LO, RFimg is below LO. We will use RF1 to denote the signal below LO and RF2 to denote the signal above LO. If RF=RF1, then RFimg=RF2. Similarly, if RF=RF2, then RFimg=RF1. RF1 and RF2 both mix down to IF=LO-RF1=RF2-LO. RF1 and RF2 mix up to RF1+LO and RF2+LO. This is illustrated in Frequency Plan for Image Rejection and Sideband Suppression..
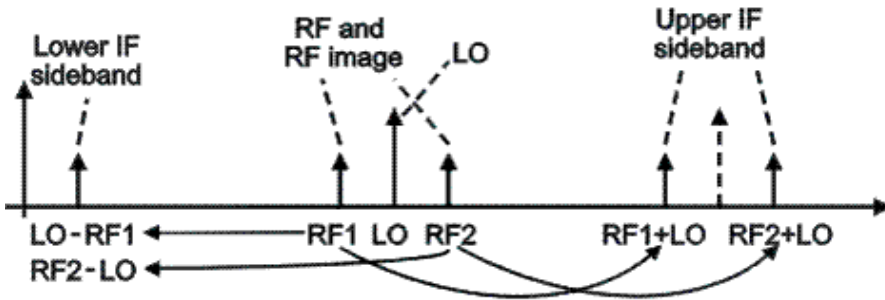


**Frequency Plan for Image Rejection and Sideband Suppression.**

We will use this to explain the five supported SideBand parameters and the use of OutputSidebandSuppression and InputImageRejection. Note that this illustration has a receiver/down-conversion bias as LO~RF. For a transmitter/up-conversion application, RF would generally be much closer to zero and LO would generally be much closer to |LO-RF| and LO+RF. However, all principles remain the same and we can use the above example to explain all SideBand options.

For SideBand=BOTH, both input images are passed through the mixer without rejection and both output sidebands are generated without suppression. This mode is the default for Mixer2. A normal circuit level mixer without external filtering operates in this manner. OutputSidebandSuppression and InputImageRejection are ignored for this mode. This is illustrated in Mixer2 image/sideband operation for SideBand=BOTH..
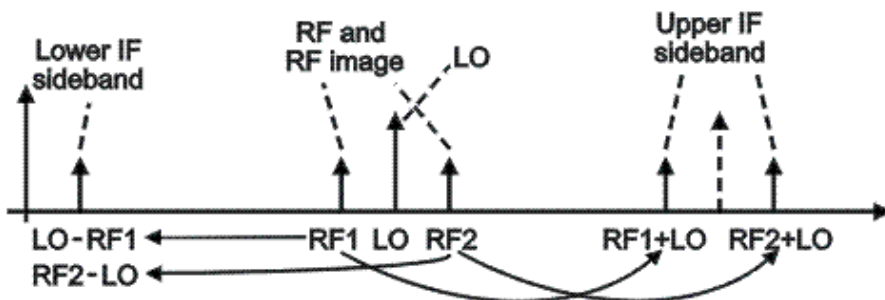


**Mixer2 image/sideband operation for SideBand=BOTH.**

For SideBand=LOWER, both input images are passed through the mixer without rejection

and the undesired upper output sideband is suppressed by the amount given by OutputSidebandSuppression relative to the desired lower output sideband. This mode is a down-conversion mode used for receiver applications. The default value for OutputSidebandSuppression is -200 dB. InputImageRejection is ignored for this mode. This is illustrated in Mixer2 image/sideband operation for SideBand=LOWER..
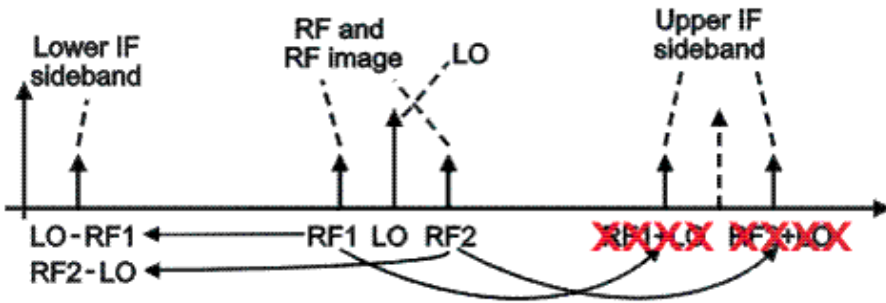


**Mixer2 image/sideband operation for SideBand=LOWER.**

For SideBand=UPPER, both input images are passed through the mixer without rejection and the undesired lower output sideband is suppressed by the amount given by OutputSidebandSuppression relative to the desired upper output sideband. This mode is an up-conversion mode used for transmitter applications. The default value for OutputSidebandSuppression is -200 dB. InputImageRejection is ignored for this mode. This is illustrated in Mixer2 image/sideband operation for SideBand=UPPER..
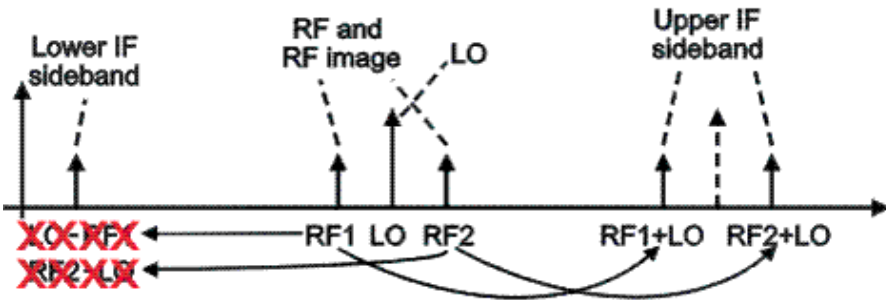


**Mixer2 image/sideband operation for SideBand=UPPER.**

For SideBand=LOWER IMAGE REJECTION, the lower input image is rejected by the amount given by InputImageRejection relative to the upper input image and both output sidebands are suppressed by the amount given by InputImageRejection. This upper sideband behavior is not intuitive but since this mode is almost exclusively used for receiver/down-conversion applications, the upper output sideband is generally not of much interest. The default value for InputImageRejection is -200 dB. OutputSidebandSuppression is ignored for this mode. This is illustrated in Mixer2 image/sideband operation for SideBand=LOWER IMAGE REJECTION..

Upper IF
sideband

Lower IF
sideband

RF and
RF image .LO

X(X-RFX ← ——————— RF1 LO RF2     XRF1+LOX RFX+XOX
RF2-LO ←
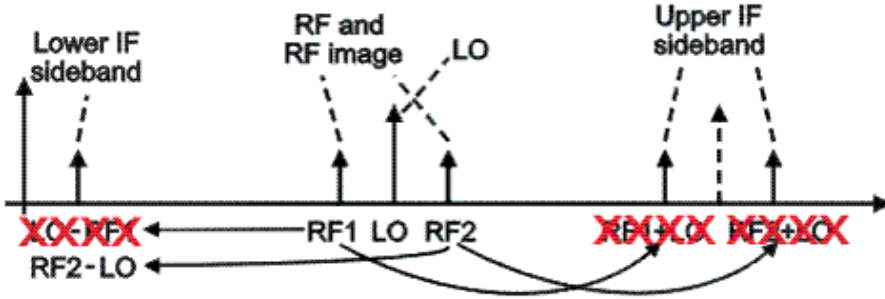
**Mixer2 image/sideband operation for**

SideBand=LOWER IMAGE REJECTION.
For SideBand=UPPER IMAGE REJECTION, the upper input image is rejected by the amount given by InputImageRejection relative to the lower input image and both output sidebands are generated without suppression. Since this mode is almost exclusively used for receiver/down-conversion applications, the upper output sideband is generally not of much interest. The default value for InputImageRejection is -200 dB. OutputSidebandSuppression is ignored for this mode. This is illustrated in Mixer2 image/sideband operation for SideBand=UPPER IMAGE REJECTION..

Upper IF
sideband

Lower IF
sideband

RF and
RF image .LO

LO-RF1 ← ——————— RF1 LO RF2     RF1+LO  RF2+LO
XRF2-LOX ←
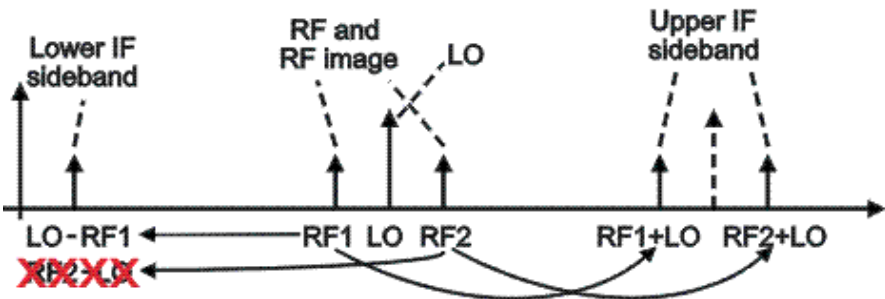
**Mixer2 image/sideband operation for**

SideBand=UPPER IMAGE REJECTION.
Other combinations of input image rejection and output sideband suppression exist which are not supported by Mixer2. None of these modes are of much practical relevance.
The InputImageRejection and OutputSidebandSuppression parameters do not support image or sideband enhancement. The input image rejection and output sideband suppression is abs (InputImageRejection) and abs (OutputSidebandSuppression). The signs of InputImageRejection and OutputSidebandSuppression do not matter.
The operation of Mixer2 for SideBand=BOTH emulates the behavior of a single mixer. The operation of Mixer2 for SideBand=LOWER and SideBand=UPPER emulates the behavior of a single mixer with a filter at the output. The operation of Mixer2 for SideBand=LOWER IMAGE REJECTION and SideBand=UPPER IMAGE REJECTION emulates the behavior of a mixing sub-system constructed from two single mixers, a phase shifter, and possibly other components.
Since circuit-level mixers pass both input images and generate both output sidebands, a practical design process will typically end up using Mixer2 with SideBand=BOTH when the

design is finalized. This, however, does not render the other four modes any less valuable. If a design uses a mixer suppressing the lower/upper output sideband, SideBand=UPPER or SideBand=LOWER will typically be used in the beginning stages of the design process. As the design is finalized and more physical realism is desired, a switch to SideBand=BOTH can be made and output filters can be added. Similarly, if a design uses a mixing sub-system rejecting the lower/upper input image, SideBand=LOWER IMAGE REJECTION or SideBand=UPPER IMAGE REJECTION will typically be used in the beginning stages of the design process. As the design is finalized and more physical realism is desired, a switch to SideBand=BOTH can be made and an actual image rejection mixing sub-system can be built from mixers with SideBand=BOTH. Such design processes take advantage of many features offered by Mixer2.

### Linear Behavior

The linear behavior of Mixer2 is described by the conversion gain ConvGain, the reverse conversion gain RevConvGain, and the nine reflection/leakage/isolation parameters (S-parameters) SPij (i,j=1,2,3).

The ConvGain parameter is the conversion gain from RF to IF. It is applied to the lower sideband |RF-LO| and the upper sideband RF+LO. Either of these sidebands can then be suppressed by OutputSidebandSuppression, as described in the "Sideband Suppression and Image Rejection" section.

If ConvGain is real, it will simply scale the output sidebands. To understand the behavior of Mixer2 when ConvGain is complex, consider a Mixer2 with SideBand=BOTH. Given an LO, we consider RF1 below LO and RF2 above LO. In a Harmonic Balance simulation, Mixer2 will down-convert to |RF1-LO|=LO-RF1 and |RF2-LO|=RF2-LO and up-convert to RF1+LO and RF2+LO. If signals with non-zero phases are applied at RF1 and RF2, the phases will transfer directly to the output tones at RF2-LO, RF1+LO and RF2+LO. However, the phase at LO-RF1 will be inverted. The reason is that RF1-LO is a negative frequency and that the simulator therefore solves for the result at the corresponding positive frequency -(RF1-LO)=LO-RF1 and then conjugates the result. This conjugation is what manifests itself in the phase reversal for the LO-RF1 output tone. It has nothing to do with Mixer2. It is a Harmonic Balance concept which can also be seen with Mixer and VMult. For Mixer2, the phase of ConvGain is applied in an analogous manner. It is applied before the mixing of RF with LO takes place and before any compression from RF to LO is applied, not after. The phase of ConvGain is therefore subjected to the mixing process which means, per above, that the phase of ConvGain will be added to the output tones at RF2-LO, RF1+LO and RF2+LO but subtracted from the output tone at LO-RF1. Mixer, on the other hand, simply applies the phase of ConvGain to all output tones regardless of any mixing taking place. The approach for Mixer2 is consistent with the underlying mixing process and appears much more appealing than that for Mixer.

The RevConvGain parameter, the reverse conversion gain from IF to RF, is similar to ConvGain, except no compression is associated with RevConvGain. Also, the phase of RevConvGain is applied after the mixing, not before. For most applications, RevConvGain will be zero.

The SPij (i,j=1,2,3) parameters describe the port reflection and port-to-port leakage/isolation for the mixer. Mixer2 is a three-port device and in line with established theory for generalized S-parameters we denote the voltages and currents at port n by vn and in and define the input and output waves at each port as:

$$an = (vn + Zn{*}in)/(2{*}\sqrt{Zn})$$
$$bn = (vn + Zn{*}in)/(2{*}\sqrt{Zn})$$

with Zn being the reference impedance for port n. This is illustrated in <u>Definition of Voltages, Currents and Waves.</u>.

The nine S-parameters SPij (i,j=1,2,3) are then defined through
b1 = SP11*a1 + SP12*a2 + SP13*a3
b2 = SP21*a1_poly + SP22*a2 + SP23*a3
b3 = SP31*a1_poly + SP32*a2 + SP33*a3
where, relative to normal generalized S-parameters, two instances of a1 have been replaced by a1_poly which is the a1 wave with polynomial compression taken into account.
SP11, SP21 and SP31 operate on a1 or a1_poly and are therefore defined at the frequencies present at the RF port. SP11*a1, SP21*a1_poly and SP31*a1_poly are therefore contributions at the RF/IF/LO port at the frequencies present at the RF port. Similarly, SP12, SP22 and SP32 operate on a2 and are therefore defined at the frequencies present at the IF port. SP12*a2, SP22*a2 and SP32*a2 are therefore contributions at the RF/IF/LO port at the frequencies present at the IF port. Similarly, SP13, SP23 and SP33 operate on a3 and are therefore defined at the frequencies present at the LO port. SP13*a3, SP23*a3 and SP33*a3 are therefore contributions at the RF/IF/LO port at the frequencies present at the LO port. All three ports have contributions at all frequencies. If SPij were normal generalized S-parameters for a three-port, bi, aj and SPij would all be defined at the same frequency. The extension to the case of multiple frequencies appears natural so S-parameter terminology was chosen for Mixer2.
SP11 is the RF reflection, SP12 is the IF to RF leakage/isolation, SP13 is the LO to RF leakage/isolation, SP21 is the RF to IF leakage/isolation, SP22 is the IF reflection, SP23 is the LO to IF leakage/isolation, SP31 is the RF to LO leakage/isolation, SP32 is the IF to LO leakage/isolation, and SP33 is the LO reflection.
ConvGain, RevConvGain and SPij (i,j=1,2,3) are specified as complex numbers. Use the functions polar(magnitude,angle), dbpolar(dB, angle), or VSWRpolar(VSWR, angle) to convert the ConvGain, RevConvGain and SPij specifications into complex numbers. For example, if a reflection/leakage is X dB, use dbpolar(-X,0).
ConvGain, RevConvGain and SPij (i,j=1,2,3) are voltage gains and not power gains. For instance, a mixer with ConvGain=polar(10,0) will scale the voltage by a factor of 10 from input to output and will therefore result in a 20 dB increase in power.
ConvGain=dbpolar(10,0), on the other hand, will result in a 10 dB increase in power.
As outlined above, SPij (i,j=1,2,3) are wave-based. The same is true for RevConvGain. ConvGain, on the other hand, is voltage-based, not wave-based. ConvGain applies to the input voltage, not the input wave. However, ConvGain is not blindly applied to the input

voltage v1 at the RF port. Doing so would make the RF to IF conversion dependent on SP11 as a modified SP11 results in a different b1 and therefore changes v1. In keeping with the measurement standards used to define mixer conversion gain at the system level, the reflected wave due to SP11 is not included. This means that if we change SP11 from 0 to something finite, we will see no change in IF output power because the reflected wave is not taken into account when mixing from RF to IF. The same holds true for S-parameters. With a fixed SP21, the RF to IF leakage does not depend on SP11. This applies to other S-parameters in an analogous manner.

To illustrate the difference between wave-and voltage-based parameters, consider a mixer with an RF source, LO source and IF termination. The RF input wave a1 is finite but the RF output wave b1 is zero. Similarly, the LO input wave a3 is finite but the LO output wave b3 is zero. The mixing results in a finite IF output wave b2 but the IF input wave a2 is zero. Now add LO to RF leakage through a finite LO_Rej1 (Mixer) or SP13 (Mixer2) parameter. This will result in a signal at the LO frequency at the RF port and generate a finite b1 wave. Assuming no reflection at the RF source, this will not change the a1 wave. The a1 wave will not have any content at the LO frequency.

Mixer has a wave-based ConvGain parameter. The RF to IF conversion is based on the a1 wave. Since the a1 wave does not change when a finite LO_Rej1 is added, neither will the RF to IF conversion. Specifically, no RF signal at the LO frequency will mix with LO to produce IF output at DC and twice the LO frequency. Thus, Mixer will produce zero DC output. Mixer2, on the other hand, has a voltage-based ConvGain parameter. The RF to IF conversion is based on the RF input voltage, not the a1 wave. Since the RF input voltage changes when a finite SP13 is added, so will the conversion. Specifically, the RF signal at the LO frequency will mix with LO to produce, absent of sideband suppression, IF output at DC and twice the LO frequency. Thus, Mixer2 will produce a finite DC output. However, note that that Mixer2 will not produce a direct feed-through IF output at the LO frequency if SP21 is finite. The S-parameters, as stated above, are wave-based, not voltage-based. They operate on a1, not v1.

## Nonlinear Behavior

The nonlinear behavior of Mixer2 is described by a number of different list- and file-based options for specifying compression. A polynomial compression model is used. For detailed information about the polynomial modeling of compression, please see *Modeling Basics* (ccsys) and *AM to PM Conversion* (ccsys) in the documentation for Amplifier2. The compression options for Mixer2 and Amplifier2 are identical, as is the underlying modeling.

Amplifier2 applies compression from its input to its output. In an analogous manner, Mixer2 applies compression from RF to IF. In addition, Mixer2 also applies the same compression from RF to LO. There is no guarantee that the compression from RF to LO is the same as that from RF to IF for an actual mixer, but applying this compression from RF to LO seems more reasonable than to not apply compression from RF to LO.

## LO Limiting

Under typical operating conditions, a circuit-level mixer's LO is saturated. This means that the RF to IF mixing process is insensitive to the actual value of the LO signal. A small fluctuation in LO power will not change the RF to IF mixing. A mixer is not a voltage multiplier. To mimic this behavior for Mixer2, the LO is limited. Mixer limits the LO based

on a filtered detector signal. Prior to Release 2004A, Mixer2 limited the LO based on the magnitude of the Hilbert transform of the LO signal. Starting in Release 2004A, Mixer2 allows limiting based on both a filtered detector signal and the magnitude of the Hilbert transform of the LO signal, with the latter being the default. Instead of the brick wall DC-only filter used for Mixer, it is now a single pole roll-off with a variable bandwidth specified by DetBW. For extremely large bandwidths (1e12 Hz and above), this switches to Hilbert transform limiting. Hilbert transform limiting (large DetBW values) is best for bandpass LO waveforms with multiple signals but does rather poorly when an LO with significant harmonics is used, as analytic signal calculations such as instantaneous magnitude do not apply to broadband signals. For an LO with significant harmonics, filtered detector signal limiting (small DetBW values) should be used.

A circuit-level mixer has an LO power level below which its behavior changes dramatically. To mimic this behavior for Mixer2, Mixer2 has a PminLO parameter which sets a limit for the effect of the LO power on the mixer's conversion gain. Mixer2 adds this power level to the limited LO power level (limited via a filtered detector signal for DetBW<=1e12 Hz and via the magnitude of the Hilbert transform for DetBW>1e12 Hz) in order to arrive at the total LO power level. Note that the model does not simply switch to using PminLO when the limited LO power level falls below PminLO. It is a gradual transition and not a threshold. When the limited LO power level is much larger than PminLO, the effect of PminLO on the conversion gain is negligible. As the limited LO power level approaches PminLO, the conversion gain will depend on the value of PminLO in a nonlinear manner. If this is undesirable, lower PminLO.

The LO power is not a fixed quantity but is computed dynamically from the signal at the LO port. This means that Mixer2's conversion gain will vary depending on the frequency content of the LO signal. Frequency content other than the primary LO signal could come from harmonics of the primary LO signal or from spurious LO signals. If the dependence of the LO power on the LO frequency content is undesired, a bandpass filter can be added to the LO path.

Filtered detector signal limiting and Hilbert transform limiting behave differently when an LO sideband enters Mixer2. Consider the case of RF=0.1 GHz, LO=1.0 GHz, |LO-RF|=0.1 GHz, and LO+RF=1.9 GHz. Assuming an RF power of -20 dBm, an LO power of 0 dBm and a conversion gain of 0 dB, we get -20 dBm IF power at both sidebands. Now add a 1.01 GHz tone at -30 dBm at LO. Filtered detector signal limiting gives zero power at 0.09 GHz and 1.89 GHz and -50 dBm power at 0.11 GHz and 1.91 GHz. Hilbert transform limiting, on the other hand, generates intermodulation products and gives -56 dBm power at 0.09 GHz, 0.11 GHz, 1.89 GHz, and 1.91 GHz. This supports the recommendation to use Hilbert transform limiting (high DetBW values, the default for Mixer2) unless significant harmonics are present at LO.

Mixer2 does not have an LO common mode leakage parameter which would allow the specification of the amount of LO common mode leakage to be present at the output. For a two-tone LO, the LO common mode leakage is the level of the sum and difference intermodulation products of the two LO tones. A workaround to realize this is to add an Amplifier2 component in the LO path, set its SOI/TOI parameters to generate the appropriate sum/difference tones, and set SP23 on Mixer2 to leak these tones from LO to IF.

### Noise

Given the minimum noise figure NFmin (real), the optimal reflection coefficient Sopt (complex), the noise resistance Rn (real), the noise reference impedance Rref (real), and the source admittance Ys (complex), the noise figure NF of an amplifier is determined by

$$Yopt = \frac{1}{Rref} \times \frac{1 - Sopt}{1 + Sopt}$$

$$Fmin = 10^{NFmin/10}$$

$$F = Fmin + \frac{Rn}{Real[Ys]} \times |Ys - Yopt|^2$$

$$NF = 10 \times \log(F)$$

Note that this is independent of the amplifier S-parameters.

The noise behavior of Amplifier2 is characterized by the four noise parameters NF, NFmin, Sopt and Rn and the reference impedance Z1 for port 1. Amplifier2 is implemented as a Noisy2Port cascaded with an SDD. The above-mentioned five parameters control the parameters for the Noisy2Port, the Noisy2Port generates a noise voltage on its output and this noise voltage is passed through the SDD in the same manner as the signal.

The noise behavior of Mixer2 mimics that of Amplifier2. Similar to Amplifier2, Mixer2 is implemented as a Noisy2Port cascaded with an SDD. The parameters NF, NFmin, Sopt, Rn and Z1 are passed to the Noisy2Port and noise is generated. The signal is not affected by the generation of noise but goes unaltered through the Noisy2Port. All parameters except NF, NFmin, Sopt and Rn are then passed to the SDD and the signal and noise goes through the SDD. The Noisy2Port part of Mixer2 is identical to that of Amplifier2 and the noise at the output of the Noisy2Port for Amplifier2 and Mixer2 will consequently be the same if the noise parameters and reference impedances are the same. The difference between Amplifier2 and Mixer2 is the SDD that follows the Noisy2Port.

NF-only mode is used for NFmin=0. This is a special case where only one noise parameter must be specified. In this case, the Noisy2Port has the parameters NFmin=NF, Sopt=0, and Rn=Z1/4 × ($10^{NF/10}$-1). The reference impedance for the noise calculation (not available on the Noisy2Port user interface) is Z1. The NFmin=0, Sopt and Rn parameters are ignored.

(NFmin, Sopt, Rn) mode is used for NFmin>0. This is a more general case than NF-only mode. In this case, the Noisy2Port has the parameters NFmin=NFmin, Sopt=Sopt, and Rn=Rn. The reference impedance for the noise calculation (not available on the Noisy2Port user interface) is Z1. The NF parameter is ignored.

Given an output noise voltage $vn$ , the single sideband noise figure NFssb and the double sideband noise figure NFdsb are given by

NFssb=10*log(( $vn^2$ / R + k × T0 × ( G1 + G2 +...)) / ( k × T0 × G1))

NFdsb=10*log(( $vn^2$ / R + k × T0 × ( G1+ G2 + ...)) / ( k × T0 × ( G1 + G2 + ...)))

where R is the output resistance, k =1.380658e-23 J/K is Boltzmann's constant, T0=290 K is the standard noise temperature, G1 is the primary power gain from the input noise frequency to the output noise frequency, and G2+... is the sum of all higher order mixing gains which mix from some input frequency to the output noise frequency. For an amplifier, G2+... is zero under small-signal operation. The same holds true for a mixer with input image rejection. For a mixer without input image rejection, G2 is significant and often equal to G1 under small-signal operation while G3+... is zero under small-signal conditions. $vn^2$ / R represents the noise added by the component(s) while k × T0 represents the noise power available from the input termination.

The documentation for Amplifier2 outlines how Amplifier2 calculates noise voltages and noise figures in various cases and compares this to the behavior of Amplifier. Due to the similarities between Amplifier2 and Mixer2, much of this material is also relevant for

Mixer2. As mentioned above, the difference between Amplifier2 and Mixer2 is not the generation of noise by the Noisy2Port but how this noise is translated through the SDD that follows. For more information about noise voltages, please see the documentation for Amplifier2.

The NF-only and (NFmin,Sopt,Rn) noise modes are covered above and many details about the computation of noise voltages for Noisy2Port are given in the Amplifier2 documentation. The last remaining aspect to be understood is the influence of SideBand, OutputSidebandSuppression and InputImageRejection on the noise voltages and noise figures.

To this end, consider a Mixer2 with a given LO. We choose RF<LO. The lower output sideband is LSB=LO-RF and the upper output sideband is USB=RF+LO. The input image for down-conversion is LSB_Img=2*LO-RF (LSB_Img mixes down with LO to give LSB) and the input image for up-conversion is USB_Img=2*LO+RF (-USB_Img mixes up with LO to give -USB). This is illustrated in <u>Frequency plan for noise computation.</u>.
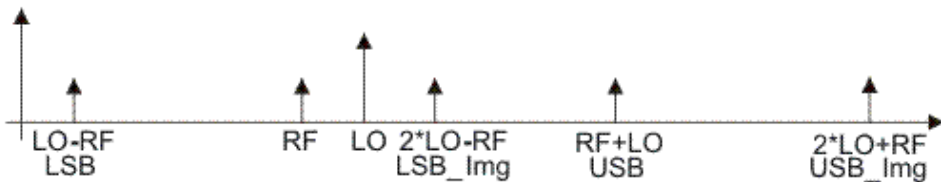


**Frequency plan for noise computation.**

We will use Mixer2 in NF-only mode. The results carry over directly to (NFmin,Sopt,Rn) mode. We operate the mixer at low RF powers. We let NF=5.00 dB. Since NF is the double sideband noise figure, we expect Mixer2 to give NFdsb=NF in all cases. This is exactly what Mixer2 gives, see <u>Noise Results</u>. NFssb, on the other hand, depends on SideBand, OutputSidebandSuppression and InputImageRejection. From the expressions for NFssb and NFdsb, we see that

NFssb=NFdsb+10*log( *(G1 + G* 2 +...) / *G* 1)
Letting all gain terms above G2 to zero and substituting NF for NFdsb, we expect
NFssb=NF+10*log(1 + *G* 2 / *G* 1)
In the following, we will explain the influence of SideBand, OutputSidebandSuppression and InputImageRejection on NFssb.

**Noise Results**

| | | | | | LSB | | USB | |
|---|---|---|---|---|---|---|---|---|
| SideBand | Rejection/dB | Suppression/dB | _NF/ dB_ | _v.noise/ pV_ | _NFdsb/ dB_ | _NFssb/ dB_ | _NFdsb/ dB_ | _NFssb/ dB_ |
| BOTH | N/A | N/A | 5.00 | 930.5 | 5.00 | 8.01 | 5.00 | N/A |
| LOWER IM REJ | 0 | N/A | 5.00 | 930.5 | 5.00 | 8.01 | N/A | N/A |
| LOWER IM REJ | -10 | N/A | 5.00 | 690.0 | 5.00 | 5.41 | N/A | N/A |
| LOWER IM REJ | -200 | N/A | 5.00 | 657.9 | 5.00 | 5.00 | N/A | N/A |
| UPPER IM REJ | 0 | N/A | 5.00 | 930.5 | 5.00 | 8.01 | N/A | N/A |
| UPPER IM REJ | -10 | N/A | 5.00 | 690.0 | 5.00 | 5.41 | N/A | N/A |
| UPPER IM REJ | -200 | N/A | 5.00 | 657.9 | 5.00 | 5.00 | N/A | N/A |
| LOWER | N/A | 0 | 5.00 | 930.5 | 5.00 | 8.01 | N/A | N/A |
| LOWER | N/A | -10 | 5.00 | 930.5 | 5.00 | 8.01 | N/A | N/A |
| LOWER | N/A | -200 | 5.00 | 930.5 | 5.00 | 8.01 | N/A | N/A |
| UPPER | N/A | 0 | 5.00 | 930.5 | N/A | N/A | 5.00 | 8.01 |
| UPPER | N/A | -10 | 5.00 | 690.0 | N/A | N/A | 5.00 | 5.41 |
| UPPER | N/A | -200 | 5.00 | 657.9 | N/A | N/A | 5.00 | 5.00 |

For SideBand=BOTH, we get NFssb=8.01 dB=NF+3.01 dB for both the lower and the upper sideband. OutputSidebandSuppression and InputImageRejection are both ignored in this case. Since this case offers no image rejection, we have G2=G1 and the expression for NFssb confirms the result by Mixer2.

For SideBand=LOWER IMAGE REJECTION and SideBand=UPPER IMAGE REJECTION, we get an NFssb value that varies from NF+3.01 dB to NF as InputImageRejection is changed from 0 dB (no rejection, same as SideBand=BOTH) to -200 dB. Since these SideBand modes are intended for down-conversion applications only, the noise figure for the upper sideband does not matter. OutputSidebandSuppression is ignored in this case. With no rejection, we have G2=G1 and the expression for NFssb confirms the result by Mixer2. With full rejection, we have G2=0 and the expression for NFssb confirms the result by Mixer2. In between, we have G2=10^(InputImageRejection/10)*G1 and expect to get NFssb=NF+10*log(1+10^(InputImageRejection/10)). For InputImageRejection=-10, this evaluates to NFssb=NF+0.41 dB and confirms the result by Mixer2.

For SideBand=LOWER, we get NFssb=NF+3.01 dB as OutputSidebandSuppression is changed from 0 dB (no suppression, same as SideBand=BOTH) to -200 dB. Since this SideBand mode is intended for down-conversion applications only, the noise figure for the upper sideband does not matter. InputImageRejection is ignored in this case. With no suppression, we have G2=G1 and the expression for NFssb confirms the result by Mixer2. As we change OutputSidebandSuppression, all we do is change the conversion gain to the upper sideband. This should have no effect on the noise in the lower sideband, our sideband of interest. Indeed, we keep having G2=G1 and the expression for NFssb confirms the result by Mixer2.

For SideBand=UPPER, we get an NFssb value that varies from NF+3.01 dB to NF as OutputSidebandSuppression is changed from 0 dB (no suppression, same as SideBand=BOTH) to -200 dB. Since this SideBand mode is intended for up-conversion applications only, the noise figure for the lower sideband does not matter.

InputImageRejection is ignored in this case. With no suppression, we have G2=G1 and the expression for NFssb confirms the result by Mixer2. As we change OutputSidebandSuppression, we change the conversion gain to the lower sideband. As for SideBand=LOWER, we might expect this to have no effect on the noise in the upper sideband, our sideband of interest. However, the input image USB_Img is above USB in this case and therefore is subjected to the conversion gain to the lower sideband which is being changed by OutputSidebandSuppression. An upper sideband mixer is, in effect, an image rejection mixer. With full suppression, we have G2=0 and the expression for NFssb confirms the result by Mixer2. In between, we have G2=10^(OutputSidebandSuppression/10)*G1 and expect to get NFssb=NF+10*log(1+10^(OutputSidebandSuppression/10)). For OutputSidebandSuppression=-10, this evaluates to NFssb=NF+0.41 dB and confirms the result by Mixer2.

Noise can be contributed and lowered in several ways. To see this, consider a down-converting mixer with a certain RF and LO. IF is |RF-LO| and RFimg is 2*LO-RF. Output noise at IF can be a result of noise at RF and RFimg mixing with LO to contribute noise at IF but can also be a result of noise at IF leaking directly through the mixer from RF or LO if it has a finite RF/LO to IF leakage. This output noise at IF can be lowered if the mixer is an image rejection mixer that rejects the signal/noise contribution at RFimg or if the mixer has no RF/LO to IF leakage. In the absence of a mixer with such properties, output noise at IF can be lowered external to the mixer by adding filters at the RF and LO ports which eliminate the input noise at RFimg and IF (RF port) and IF (LO port). Eliminating the noise at RF is much harder since adding a filter at RF will also eliminate the desired signal at RF.

### Notes/Equations

1. The following table describes the sideband parameter mapping to the actual integer values:

| Parameter | Integer value |
| --- | --- |
| LOWER | -1 |
| BOTH | 0 |
| UPPER | 1 |

2. Mixer2 is introduced as a replacement for Mixer for anything but frequency conversion AC analysis (see note 3). To change an existing Mixer component to a Mixer2 component, change the name from Mixer to Mixer2 on the schematic. Many parameters for the two models are the same and Mixer2 will adopt the values for Mixer, making parameter re-entry unnecessary. The only exception is that the parameters SideBand, GainCompType and ReferToInput will take their default values BOTH, LIST and OUTPUT, respectively, regardless of the values these parameters had for Mixer. The Mixer parameters ImageRej, LO_Rej1, LO_Rej2, RF_Rej, S11, S22 and S33 correspond to the Mixer2 parameters OutputSidebandSuppression, SP13, SP23, SP21, SP11, SP22 and SP33 and will have to be re-entered. In addition, Mixer2 has the new parameters InputImageRejection, SP12, SP31, SP32, DetBW, AM2PM, PAM2PM and ClipDataFile whose values cannot be deduced from Mixer and will therefore take their default values. Also, the Mixer display settings will be ignored. Mixer2 will adopt its default settings, displaying SideBand and ConvGain. Major Differences between Mixer and Mixer2. summarizes the major differences between Mixer and Mixer2

**Major Differences between Mixer and Mixer2.**

| Mixer | Mixer2 |
|---|---|
| Image rejection not supported | Image rejection supported |
| Reverse conversion gain not supported | Reverse conversion gain supported |
| Some leakage terms supported | All leakage terms supported |
| Real/imaginary polynomial fit | Magnitude/phase polynomial fit |
| AM to PM not supported | AM to PM supported for all magnitude modes |
| One type of LO limiting | More flexible LO limiting |
| Complex ConvGain leads to non-physical behaviors | Complex ConvGain leads to physical behaviors |
| FCAC analysis supported | FCAC analysis not supported |
| For large harmonic balance and circuit envelope simulations Mixer2 may be slower than Mixer. | |

3. For a tutorial example of how to use Mixer2 in various cases, see examples/Tutorial/Mixer2_Example_wrk. For other examples of how to use Mixer2, search the ADS examples for the Mixer2 component.
4. Mixer supports frequency conversion AC (FCAC) analysis for small-signal AC or S-parameter analysis, while Mixer2 does not. This capability allows small-signal frequency analysis traditionally done at only one frequency to be somewhat extended to deal with more than one frequency. It is not as accurate as harmonic balance analysis. If FCAC analysis is needed, use Mixer. If not, use Mixer2.
5. Mixer2 does not support complex reference impedances.
6. Mixer2 passes dc.
7. Mixer2 is not recommended for baseband envelope applications.
8. Mixer2 may be slower than Mixer for large harmonic balance and circuit envelope simulations.
9. When using Mixer2 in transient simulations, Mixer2 gives a warning about non-causality of the H(2), H(15) and H(18) weighting functions. Please ignore this warning.
10. When setting up simulations with Mixer2, make sure to avoid colliding tone issues where multiple mixing products map to the same frequency. Setups with colliding tones should be changed so colliding tones are eliminated.
11. This note describes Mixer2 operation with the SOI and TOI parameters set. SOI and TOI are used for specifying the second-order and third-order intercept points IP2 and IP3.

    The general equation for the nth order intercept point IPn is IPn=(n*P1-Pn)/(n-1) where P1 is the power level of the first-order tone and Pn is the power level of the nth-order tone. The latter power level, however, is not unique. It can be based on either harmonics or intermods, and the two will generally not be the same. If the formula IPn= (n*P1-Pn)/(n-1) works for one type of second- and third-order tone, it will not work for the other. The industry standard for IPn is based on intermods, and the same goes for the SOI and TOI parameters for Mixer2. IPn must therefore be validated using a two-tone setup, not a one-tone setup. Note that IPn is defined at a low power level. If IPn is computed at a power level where either P1 or Pn deviate from their low-power values, the results will be in error. Note that it is not enough that the fundamental tone varies linearly. IPn is computed based on the fundamental tone and a higher-order intermod so one must ensure that the higher-order intermod is also linear or IPn will change. Also, note that IPn is defined under output-matched conditions.

    While the basic definition of the power levels used for computing IPn are the same for

an amplifier and a mixer, this is not the case for the frequencies at which these power levels are evaluated.

For an amplifier with two input tones at f1 and f2 (assume f1<f2) at the same power level, a two-tone harmonic balance simulation will result in first-order tones at f1 and f2, second-order intermod products at f2-f1 and at f1+f2, and third-order intermod products at 2 × f1-f2 (will be smaller than f1 by f2- f1) and 2 × f2-f1 (will be greater than f2 by f2-f1). For computing IP2=SOI, we can use the power levels at f1 and f2-f1 or we can use the power levels at f2 and f1+f2. For computing IP3=TOI, we can use the power levels at f1 and 2 × f1-f2 or we can use the power levels at f2 and 2 × f2-f1.

For a mixer, everything is the same except that all frequencies are being shifted down (down-converting mixer) or up (up-converting mixer) by the LO frequency LO due to mixing. Therefore, all these frequencies will either have LO subtracted or added. Note that this means that from a harmonic balance point of view, IP2=SOI is calculated off of second- and third-order intermod products and IP3=TOI is calculated off of second- and fourth-order intermod products. For computing IP2=SOI, we can use the power levels at f1+/-LO (second-order as far as the harmonic balance simulation is concerned) and at f2-f1+/-LO (third-order as far as the harmonic balance simulation is concerned) or we can use the power levels at f2+/-LO (second-order as far as the harmonic balance simulation is concerned) and at f1+f2+/-LO (third-order as far as the harmonic balance simulation is concerned). For computing IP3=TOI, we can use the power levels at f1+/-LO (second-order as far as the harmonic balance simulation is concerned) and at 2 × f1-f2+/-LO (fourth-order as far as the harmonic balance simulation is concerned) or we can use the power levels at f2+/-LO (second-order as far as the harmonic balance simulation is concerned) and at 2 × f2-f1+/-LO (fourth-order as far as the harmonic balance simulation is concerned).

12. When the RF port is excited with two tones, the down/up-converted sum/difference frequencies are present at the IF port. What about the sum/difference frequencies themselves? The difference frequency can be very important for a zero-IF down-converting mixer. The answer is that the sum/difference frequency level at the IF port depends on SP21 and SOI. For SP21=0, the sum/difference frequency level is zero. For finite SP21 values, the sum/difference frequency level depends on SOI. The sum/difference frequency level will be zero at zero input power and will rise linearly with the expected 2:1 slope as the input power level is increased. Consequently, the sum/difference frequency level gets more and more significant as the input power gets closer and closer to SOI. At a high enough input power level, the sum/difference frequency level compresses. If SOI is excessively high, the sum/difference frequency level remains low for all realistic input powers. The shape of the curve is determined by SOI and can be scaled by varying SP21. The sum/difference frequency level will experience a sharp increase when the mixer starts hard limiting, regardless of the SP21 and SOI values. For an explanation of why this happens, see the "Modeling Basics" section of the Amplifier2 documentation. All this is also true at the LO port, except that SP31 controls this level.

13. An S2D file typically consists of an ACDATA block containing small-signal information and a GCOMPi block (i=1,...,7) containing compression information. For Mixer2, the ACDATA block is ignored and the S-parameters specified on the Mixer2 component are used. Similarly, any NDATA blocks containing noise data are ignored by Mixer2.

14. When an S2D file contains gain compression data at more than one frequency, the GainGompFreq must be set to one of the frequencies in the S2D file to identify the data to be used. It is imperative that GainCompFreq be set to one of the frequencies in the S2D file as no interpolation or extrapolation between gain compression data at

different frequencies can be performed.

15. When an S2D file has a power range that exceeds that of a simulation, a choice must be made for the power range used for fitting. Assume an S2D file covers -30 dBm to 30 dBm but that a simulation is carried out from -10 dBm to 10 dBm. In this case, a choice must be made as to whether the polynomial fitting of S2D data is done over the power range -30 dBm to 30 dBm or -10 dBm to 10 dBm. In the former case, the fitting may be inaccurate as the polynomial must cover a large power range that could hold a lot of variations. This is undesirable. However, the advantage of this approach is that the results we get when simulating from -10 dBm to 10 dBm are a subset of what we would have gotten in that interval had we simulated from -30 dBm to 30 dBm. In the latter case, the fitting is much more accurate as the fitting is done over a much smaller power range which presumably holds a lot less variation. This is desirable. However, the problem with this approach is that the results between -10 dBm and 10 dBm will be different for a simulation done from -30 dBm and 30 dBm rather than from -10 dBm and 10 dBm since the polynomial coefficients change as we change the power range of the simulation. ADS does the former. It fits a polynomial to the whole S2D file, not just the subset for which the simulation is carried out. To change the fitting in a power range, it is not enough to change the power range of the simulation. To change the fitting, one must modify the S2D file. The S2D file power range, not the simulation power range, dictates the fitting power range. This is relevant in the following where we shall discuss different fittings in different power ranges.

16. A typical Pout (output power) vs. Pin (input power) curve consists of a linearly increasing region, a transition region and a saturation region. Another way of thinking of this is that typical Pout-Pin vs. Pin curve consists of a flat region, a transition region and a linearly decreasing region.
When the saturation region is made larger and larger, the fitting approach adopted by Mixer2 (polynomial fitting, odd order terms, order dependent upon the number of data points in the S2D file, max order 9) will tend to produce fitted curves which ring/oscillate more and more at higher powers. Mild ringing is often tolerable and might not even be noticed but if the transition region becomes too large it can make the results useless. To alleviate the problem, reduce the size of the saturation region to the minimum needed and leave no extra points in the S2D file. If the results are still not satisfactory, make sure ClipDataFile is set to yes and reduce the saturation region even more, relying on Mixer2 extrapolation. If the results are still not satisfactory, try breaking the S2D file into two files and simulate the problem in two steps.
If fitted results do not accurately match the data in the S2D file and it is uncertain if this ringing problem is the cause, the problem is very easy to exaggerate. Simply extend the GCOMP7 block of the S2D file with a large flat region (more input powers, saturated output power, saturated output phase) and re-simulate. If the ringing problem is the cause, the results should get worse.

17. The S2D file capability is a legacy from OmniSys. OmniSys used GComp1-GComp7 data items for specifying gain compression. *Gain Compression Data for OmniSys and ADS* summarizes the gain compression data for OmniSys and ADS. Refer to OmniSys Parameter Information for OmniSys parameter information. GComp1-GComp6 can be specified by using the corresponding ADS gain compression parameters and setting GainCompType=LIST or they can be contained in an S2D format setting GainCompType=FILE.

| OmniSys | ADS |
|---|---|
| GComp1: IP3 | TOI |
| GComp2: 1dBc | GainComp=1dB |
| GComp3: IP3, 1dBc | TOI |
| GComp4: IP3, Ps, GCS | TOI |
| GComp5: 1dBc, Ps, GCS | GainComp=1dB |
| GComp6: IP3, 1dBc, Ps, GCS | TOI |
| GComp7 | GainCompType=FILE |



**OmniSys Parameter Information**

# Mixer (First RF System Mixer, Polynomial Model for Nonlinearity)

## Symbol
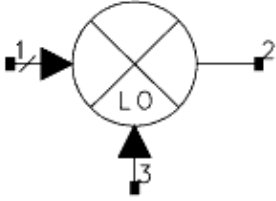


## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| SideBand | Produce UPPER, LOWER, or BOTH sidebands at output port. (See note 12) | None | UPPER |
| ImageRej | Image rejection at output with respect to fundamental | dB | None |
| LO_Rej1 | LO to Input Port Rejection | dB | None |
| LO_Rej2 | LO to Output Port Rejection | dB | None |
| RF_Rej | Input to Output Port Rejection | dB | None |
| ConvGain | Conversion Gain, use x + j × y, polar(x,y), dbpolar(x,y) for complex value; see *note 2*) | None | dbpolar(0,0) |
| S11 | Forward Reflection Coefficient, use x + j × y, polar(x,y), dbpolar(x,y), vswrpolar(x,y) for complex value; see *note 2*) | None | polar(0,0) |
| S22 | Reverse Reflection Coefficient, use x + j × y, polar(x,y), dbpolar(x,y), vswrpolar(x,y) for complex value; see *note 2*) | None | polar(0,180) |
| S33 | LO Port Reflection Coefficient, use x + j × y, polar(x,y), dbpolar(x,y), vswrpolar(x,y) for complex value; see *note 2*) | None | 0 |
| PminLO | Minimum LO Power before starvation | None | None |
| NF | Double sideband noise figure | dB | None |
| NFmin | Minimum double sideband noise figure at Sopt | dB | None |
| Sopt | Optimum source reflection for minimum noise figure, use x + j × y, polar(x,y), dbpolar(x,y) for complex value | None | None |
| Rn | Equivalent noise resistance | | |
| Z1 | Reference impedance for port 1 (real or complex number) | | |
| Z2 | Reference impedance for port 2 (real or complex number) | | |
| Z3 | Reference impedance for port 3 (real or complex number) | | |
| ImpNoncausalLength | Non-causal function impulse response order | Integer | None |
| ImpMode | Convolution mode | Integer | None |
| ImpMaxFreq | Maximum frequency to which device is evaluated | None | |
| ImpDeltaFreq | Sample spacing in frequency | None | |
| ImpMaxOrder | Maximum allowed impulse response order | Integer | None |
| ImpWindow | Smoothing window | Integer | None |
| ImpRelTol | Relative impulse response truncation factor | None | None |
| ImpAbsTol | Absolute impulse response truncation factor | None | None |

### Range of Usage

NF ≥ 0 dB
NFmin > 0
0 < Sopt < 1
0 < Rn
GainCompFreq > 0
ConvGain > 0

### Gain Compression Parameters

| Name | Description | Units | Default |
|---|---|---|---|
| GainCompType | Gain compression type: LIST=use model gain compression specifications; FILE=use file-based gain compression data | None | LIST |
| GainCompFreq | Reference frequency for gain compression (if gain compression is described as a function of frequency) | | |
| ReferToInput | Specify gain compression with respect to input or output power of device | None | OUTPUT |
| SOI | Second order intercept | dBm | None |
| TOI | Third order intercept | dBm | None |
| Psat | Power level at saturation | dBm | None |
| GainCompSat | Gain compression at Psat | dB | 5.0 |
| GainCompPower | Power level in dBm at gain compression specified by GainComp | dBm | None |
| GainComp | Gain compression at GainCompPower | dB | 1 dB |
| GainCompFile | Filename for gain compression data in S2D file format | None | None |

## Range of Usage for Gain Compression Parameters

When specifying gain compression using model parameters, only certain combination of parameters will produce stable polynomial curve fitting. Recommended parameter combinations are listed here.

> ⓘ **Note** If unrealistic parameter values are used, the polynomial will become unstable, resulting in oscillations.

- Third-order intercept and 1dB gain compression parameters:
  TOI, GainCompPower, with GainComp=1dB.
  Range of validity: TOI > GainCompPower + 10.8.
- Third-order intercept and power saturation parameters:
  TOI, Psat, GainCompSat.
  Range of validity: TOI > Psat + 8.6.
- 1dB gain compression and power saturation parameters: GainCompPower, with GainComp=1dB, and Psat, GainCompSat.
  Range of validity: Psat > GainCompPower + 3.
- Third-order intercept, 1dB gain compression and power saturation parameters:
  TOI, GainCompPower with GainComp=1dB, and Psat, GainCompSat.
  Range of validity: Psat > GainCompPower +3, TOI > GainCompPower + 10.8
- Second-order intercept and third-order intercept parameters: SOI, TOI.

## Notes/Equations

1. The Mixer component is similar to Mixer2. The key difference is that Mixer supports frequency conversion AC analysis or FCAC analysis for small-signal AC or S-parameter analysis, while Mixer2 does not. This capability allows small-signal frequency traditionally done at only one frequency to be somewhat extended to deal with more than one frequency. In terms of convergence, Mixer2 is typically more robust than Mixer, as the power detection is implemented differently.
2. If NFmin, Sopt, and Rn are used to characterize noise, the following relation must be satisfied for a realistic model.

$$\frac{Rn}{Zo} \geq \frac{To(Fmin-1)|1+Sopt|^2}{T\,4} \frac{(1-|S_{11}|^2)}{|1-Sopt\,S_{11}|^2}$$

A warning message will be issued if Rn does not meet this criterion. If the noise parameters attempt to describe a system that requires negative noise (due to Rn being too small), the negative part of the noise will be set to zero and a warning message will be issued.

3. Use the function polar(mag,ang) or dbpolar(dB, ang), or VSWR polar(VSWR, ang) to convert these specifications into a complex number.

4. For an S-parameter or a noise figure sinusoidal ripple, use the function ripple (mag, intercept, period, variable); for example ripple(0.1, 0, 10 MHz, freq).
   Example: S21=dbpolar(10+ripple( ),0.)

5. Z1, Z2 and Z3 the reference impedance parameters for ports 1, 2 and 3 are used in conjunction with the parameters S11/S21/S12/S22. This is because S-data is always used with respect to a particular reference impedance.

6. This model passes DC in the sense that a DC source at the RF input passes through the mixer to give a signal at the IF output.

7. In harmonic balance simulations, the PminLO parameter sets a threshold for the effect of the LO power on the mixer's conversion gain. The default value of PminLO is -100 dBm. The mixer will provide the expected conversion gain if the LO power is significantly greater (~20 dB) than the value of PminLO. If the LO power is less than this amount, the mixer's conversion gain will deteriorate in a nonlinear fashion.

8. Gain compression can be specified by using the gain compression model parameters, or this information can be contained in an S2D format file. All S2D gain compression types are supported by this model. Gain compression types 1 through 6 can also be described using the gain compression model parameters. Gain Compression 7 information must be contained in an S2D file. The GainCompType parameter instructs the model where to look for this data-in an S2D file or use model parameters.
   For S2D data file format information refer to *Working with Data Files* (cktsim).

9. Mixer operation with the SOI and TOI parameters is described in this note.
   First consider an amplifier. Consider two input tones at f1 and f2 (assume f1<f2) and at the same power level. Do a 2-tone HB simulation. The output will have first-order tones at f1 and f2. It also have second-order intermod products at f2-f1 and at f1+f2. It will also have third-order intermod products at 2 × f1-f2 (will be smaller than f1 by f2- f1) and 2 × f2-f1 (will be greater than f2 by f2-f1). Drawing a picture of this frequency plan can be helpful. Based on these tones we can calculate the SOI and TOI of the amplifier. The general equation is IPn= (n × P1-Pn)/(n-1) where P1 is the power level of the first-order tone and Pn is the power level of the nth-order tone. For SOI, we can use the power levels at f1 and f2-f1 or we can use the power levels at f2 and f1+f2. For TOI, we can use the power levels at f1 and 2 × f1-f2 or we can use the power levels at f2 and 2 × f2-f1.
   For a mixer, everything is the same, except that everything is being shifted down (down- converting mixer) or up (up-converting mixer) by the LO frequency fLO due to mixing. Therefore, all these frequencies will either have fLO subtracted or added. Note that this means that from an HB point of view, we're calculating SOI off of second- and third-order intermod products and calculating TOI off of second- and fourth-order intermod products. For example, TOI can be calculated from a P1 at f1-fLO (second- order as far as the HB simulation is concerned) and a Pn at 2 × f1-f2-fLO (fourth-order as far as the HB simulation is concerned).
   Also, note that the mixer must be output-matched in order to validate SOI and TOI. For more information about SOI and TOI, refer to the Amplifier2 component documentation.

10. Frequency Conversion AC (FCAC) analysis requires a single frequency at every node of a circuit. If Sideband=BOTH is specified for Mixer, the simulator will use SideBand=LOWER for FCAC analysis.

11. When Mixer is used in Transient analysis for single sideband applications, it is recommended that the user set Sideband=BOTH and insert a high-order filter to suppress the undesired sideband.

12. The following table describes the sideband parameter mapping to the actual integer values:

| Parameter | Integer value |
|-----------|---------------|
| LOWER | -1 |
| BOTH | 0 |
| UPPER | 1 |

# MixerWithLO (Mixer with Internal Local Oscillator)

## Symbol
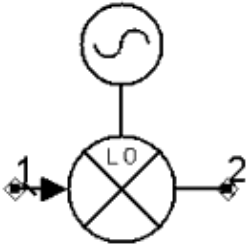


## Parameters

| Name | Description | Units | Default |
|---|---|---|---|
| ZRef | Reference impedance for all ports | Ohm | 50 |
| DesiredIF | Desired mixer IF output signal (used for Budget analysis only) | None | RF plus LO |
| ConvGain | Conversion gain, use x + j × y, polar(x,y), dbpolar(x,y) for complex value | None | dbpolar(0,0) |
| SP11 | Port 1 reflection, use x + j × y, polar(x,y), dbpolar(x,y), vswrpolar(x,y) for complex value | None | polar(0,0) |
| SP22 | Port 2 reflection, use x + j × y, polar(x,y), dbpolar(x,y), vswrpolar(x,y) for complex value | None | polar(0,180) |
| NF | Double sideband noise figure | dB | 1 |
| ReferToInput | Specify gain compression with respect to input or output power of device: OUTPUT=refer to output; INPUT=refer to input | None | OUTPUT |
| SOI | Second order intercept | dBm | 1000 |
| TOI | Third order intercept | dBm | 1000 |
| LO_Freq | Mixer local oscillator frequency | GHz | 1 |

## Range of Usage

ZRef > 0
DesiredRF = 0 (RF plus LO), 1 (RF minus LO)
SPij > 0 (ij=11; 22)
NF ≥ 0 dB
LO_Freq > 0

## Notes/Equations

1. MixerWithLO is a two port mixer model with internal oscillator.
2. This component is a schematic design that uses internal models for Mixer2 and I_1Tone. The schematic design is shown here:
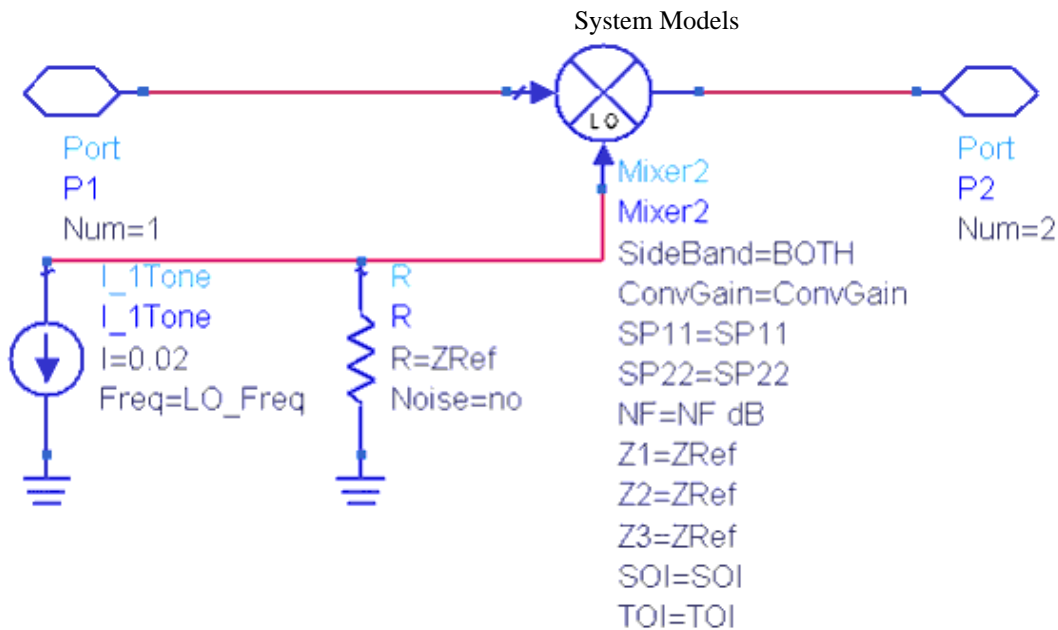
3. See documentation for the internal components Mixer2 and I_1Tone for their usage detail.

4. The internal Mixer2 model has parameter SideBand=BOTH. This MixerWithLO outputs both the RF+LO and RF-LO sidebands. However, there is no RF or LO leakage.

5. Internal to the Mixer2 model, the NF parameter is used to set the Mixer2 parameters for NFmin=NF, Sopt=0, and Rn=max( ZRef/4 × (10^{NF/10}-1), tinyreal), where *tinyreal* is an internal simulator value for the smallest real value allowed.

6. For use with Budget Analysis, the DesiredRF parameter identifies which of the output sidebands is to be tracked for Budget measurements.

7. Regarding SOI and TOI, the MixerWithLO must be output-matched in order to validate SOI and TOI.
   By default, SOI and TOI are blank resulting in no nonlinearity. TOI may be set without setting SOI. However, if SOI is set, then setting TOI is required. For more information about SOI and TOI, refer to the Mixer2 component documentation.

8. Examples demonstrating this model use in a Budget analysis are located in the ADS Examples area at: $HPEESOF_DIR/examples/Tutorial/RF_Budget_Examples_wrk. See Budget_Mixer.

# OpAmp (Operational Amplifier)

## Symbol
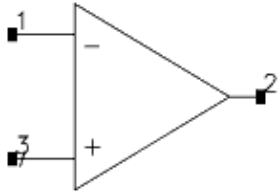


## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Gain | Open loop DC gain of amplifier | dB | 100 |
| CMR | Common mode rejection ratio | dB | None |
| Rout | Output resistance | Ohm | 100 |
| RDiff | Differential input resistance | MOhm | 1 |
| CDiff | Differential input capacitance | F | 0 |
| RCom | Common mode input resistance | MOhm | 1 |
| CCom | Common mode input capacitance | F | 0 |
| SlewRate | Signal slew rate | V/sec | 1e+6 |
| IOS | Input offset current | A | 0 |
| VOS | Input offset voltage | V | 0 |
| BW | Gain bandwidth product (unity gain bandwidth) | MHz | 1 |
| Pole1 | Dominant pole frequency (overides BW parameter) | | |
| Pole2 | Additional higher order pole frequency | | |
| Pole3 | Additional higher order pole frequency | | |
| Pole4 | Additional higher order pole frequency | | |
| Pole5 | Additional higher order pole frequency | | |
| Zero1 | Feed forward zero frequency | | |
| Inoise | Input spectral noise current | A/sqrt(Hz) | 0 |
| Vnoise | Input spectral noise voltage | V/sqrt(Hz) | 0 |
| VEE | Negative supply voltage | V | 15 |
| VCC | Positive supply voltage | V | 15 |
| DeltaVEE | Delta difference between saturated voltage and negative supply voltage (see note 4) | V | 0.3 |
| DeltaVCC | Delta difference between saturated voltage and positive supply voltage (see note 4) | V | 0.3 |

## Range of Usage

RDiff > 0
CDiff > 0

RCom > 0
CCom > 0
by default:
CMR = ∞
Pole2 = ∞
Pole3 = ∞
Pole4 = ∞
Pole5 = ∞
Zero1 = ∞

### Notes/Equations

1. The BW parameter is the GainBandwidth product, i.e. it is the frequency at which the gain is unity or 0 dB. Pole1 is the basic amplifier pole and corresponds to the frequency where the gain starts sloping downward.
   BW and Pole1 can be specified simultaneously; however, if both are entered, Pole1 will override BW, and Pole1 must then be entered as BW/Gain. The Gain parameter is the open loop gain of the opamp and it must be converted out of dB only for use in setting Pole1, i.e., Gain=$10^{Gain\_dB/20}$.
2. To match the phase shift from the data sheet, adjust the values of Pole2 through Pole5.
3. Zero1 is used for operational amplifiers with feed-forward or lead-lag compensation networks.
4. Output voltage is generally less than the rail voltage (VCC and VEE). Use DeltaVCC and DeltaVEE to specify the difference between the rail voltage and actual output voltage. For example, if VCC is +5V and the positive output is +4.5V, set DeltaVCC to 0.5V.
5. This opamp is a nonlinear model. If your circuit cannot achieve convergence using this model, use the OpAmpIdeal linear model.
6. The relationship between input and output voltages is given in the equation:
   Im × tanh(Vin/Im)=Vout/A0 - 10 × (Vclip-Vout) + d/dt (Vout × Tau1/A0)
   where
   A0 is open loop DC gain
   Vclip = Vout as long as it is not limiting
   Im = SlewRate × Tau1/A0
   Tau1=A0/2/π/BW when Pole1=0, otherwise Tau1=1/2/π/BW
7. In releases before ADS 2008, the OpAmp component contributed noise from the following noise contributors: AMP1.SRC1 and AMP1.SRC2. Starting with ADS 2008, only one noise contributor, called AMP1, is published.

# OpAmpIdeal (Ideal Operational Amplifier)

## Symbol
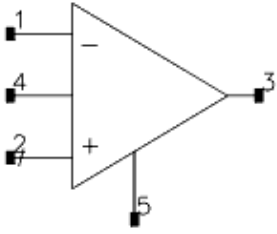


## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Gain | Complex Voltage Gain, use x + j × y, polar(x,y), dbpolar(x,y) | None | 100000 |
| Z1 | Input impedance, inverting terminal | | |
| Z2 | Input impedance, non-inverting terminal | | |
| Z3 | Output impedance | | |
| Z4 | Leakage impedance, inverting to non-inverting terminal | | |
| Freq3db | Frequency at which gain magnitude is down by 3dB | | |
| Delay | Time delay associated with gain | sec | 0 |

## Range of Usage

by default:
Z1 = ∞
Z2 = ∞
Z3 = 0
Z4 = ∞
Freq3db = ∞

## Notes/Equations

1.
$$Vs = (V^+ - V^-)Gain \times \frac{e^{-j2\pi F Delay}}{1 + j\frac{F}{Freq\ 3dB}}$$

$$Vs = (V^+ - V^-)Gain\ (for\ f=0)$$

   where F is the simulation frequency
2. OpAmpIdeal is a noiseless component.
3. The recommendation is to use a Gain value no greater than 1e10, or 200 dB for practical purposes.

## Equivalent Circuit
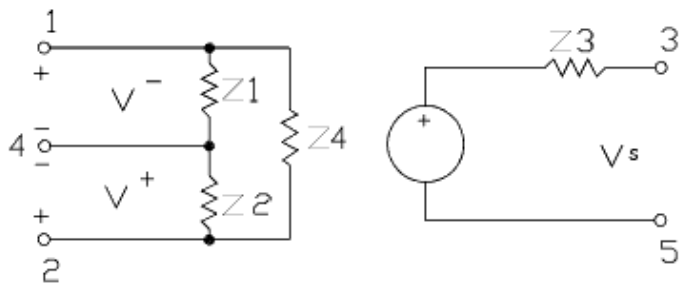
# VMult (Voltage Multiplier)

## Symbol
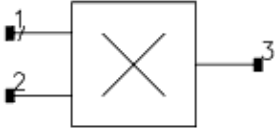


## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| R1 | Reference resistance for port 1 | Ohm | 50 |
| R2 | Reference resistance for port 2 | Ohm | 50 |
| R3 | Reference resistance for port 3 | Ohm | 50 |
| L31 | Loss in dB, pin 1 to pin 3; for linear analysis only | dB | 0 |
| L32 | Loss in dB, pin 2 to pin 3; for linear analysis only | dB | 0 |
| Linear | Use linear analysis mode: yes or no | None | no |

## Notes/Equations

1. VMult uses reference input and output impedances. Its output voltage is equal to the product of its two input voltages.
2. If Linear is set to yes,

$$V_3 = \frac{A_{31}V_1 + A_{32}V_2}{R_3}$$

A31 and A32 are the L31 and L32 losses converted from dB to regular values.
If Linear is set to no, V3 is formed from V1 and V2 through an ideal mixing process; for information on this process refer to the SDD example, *Ideal Mixer* (modbuild).

# Filters for System Models

## Introduction

The *Filters - < filter type >* and *System - < device type >* palettes contain two fundamentally different types of behavioral system models.

*Filters* , *System - Amps & Mixers* , and *System - Mod/Demod* can be classified as *tops-down system models* that support a tops-down system design flow where model behaviors are characterized by a small number of independent parameters such as frequency, power and load. They are often referred to as *parameter-based behavioral models* .
*System - Data Models* can be classified as *bottoms-up system models* that support a bottoms-up verification flow where model behaviors are extracted from a simulation (or measurement) of a transistor-level circuit. They are often referred to as *data-based behavioral models* .

The *parameter-based behavioral models* typically provide superior speed relative to the data-based behavioral models with both of these being vastly superior to a brute-force transistor-level simulation.

The *data-based behavioral models* typically provide superior accuracy relative to the *parameter-based behavioral models* as they capture actual behaviors of implemented circuit components and not just design specifications.

The differences between *parameter* - and *data-based behavioral models* justify a palette emphasis on flow (all data-based behavioral models grouped together) rather than functionality (all amplifiers, mixers, modulators, demodulators etc. grouped together).

The use model for *parameter-based behavioral models* is to simply set a series of parameters prior to using the model. The use model for *data-based behavioral models* is described in *System Data Models* (ccsys).

## Filter Categories

The filter component libraries contains filters in eight response categories: Butterworth, Chebyshev, Elliptic, Gaussian, Bessel-Thompson, Raised-Cosine, Pole-Zero, and Polynomial. Lowpass, highpass, bandpass, and bandstop filters are available in each category.

Butterworth, Chebyshev, and Elliptic filters have good selectivity but poor group delay flatness; Bessel-Thompson and Gaussian filters have good delay flatness but poor selectivity. Raised-Cosine filters are uni-directional ideal Nyquist filters for bandlimiting digital signals. Pole-Zero and Polynomial filters allow users to define arbitrary response shapes.

Except for the Raised-Cosine category, the filter S-parameters are calculated based on standard filter synthesis theory [1]. $S_{21}$ and $S_{12}$ include losses specified by unloaded

quality factor ($Q_u$) and insertion loss IL, where applicable. The assumption is made that filter pole predistortion is used to preserve the specified frequency response in the presence of losses [2]. However, $S_{11}$ and $S_{22}$ neglect losses, an approximation that causes little error for realizable filters.

The basic nature of the response of a lowpass, highpass, bandpass, and bandstop filter is illustrated below. The illustrations include certain filter parameter definitions: $f_{pass}$, $f_{stop}$, A$_{pass}$, and A$_{stop}$ for lowpass/highpass filters, and $f_{center}$, BW$_{pass}$, BW$_{stop}$, A$_{pass}$, and A$_{stop}$ for bandpass/bandstop filters. (Note that all filters do not have all of these parameters.)

In addition to these filter parameters, Gaussian and Bessel filters (XXX_Gaussian and XXX_Bessel (XXX = LPF, HPF, BPF, or BSF) have a group delay parameter GD$_{pass}$. This parameter is motivated by the fact that a signal experiences a delay when passing through a filter. Calculated as the negative of the derivative of the phase response with respect to frequency, this delay (the group delay) will be frequency dependent for filters with a non-linear phase response. In other words, the group delay at a given frequency specifies the delay experienced by a group of sinusoidal components all having frequencies within a narrow interval around that frequency. A filter whose time-domain impulse response is symmetric around t=0 must be subjected to a certain finite shift (the group delay) in order to maintain (at least some resemblance of) causality when a transient simulator truncates its impulse response below zero. Different time-frequency characteristics for different filters can lead to different requirements for this group delay.

For LPF_Bessel and LPF_Gaussian, the group delay parameter GD$_{pass}$ is defined as the group delay at the passband edge frequency relative to that at zero frequency (this is illustrated below). For highpass, bandpass, or bandstop filters, the definition is the same except that the group delay at infinity (highpass) or at the center frequency (bandpass, bandstop) is used as the reference. GD$_{pass}$ cannot drop below zero and cannot exceed one; its default value is 0.9. The group delay value has a significant effect on filter order and therefore filter rejection. Larger GD$_{pass}$ values will result in longer delays and larger filter orders; smaller GD$_{pass}$ values will result in smaller delays and smaller filter orders. If your Gaussian or Bessel filter provides less out-of-band rejection than you expect, try increasing the GD$_{pass}$ parameter.

There are two ways of dealing with filter order, N:

- Leave the filter order at zero and specify the parameters characterizing the behavior of the filter, per the illustrations below. Given these specifications, the program will calculate and report filter order N which meets these specifications. Because N must be an integer, the calculated order N exceeds the specifications in most cases. N is capped at the upper value of 15. For Chebyshev filters, the calculated filter order N must be an odd number to ensure filter symmetry.
- Alternatively, specify filter order N explicitly. If a non-zero N is specified, it will overwrite the filter specifications. The filter response is simply calculated based on the specified order N. As above, N must be an integer and will be capped at 15. And, N must be odd for Chebyshev filters.

GD(f)

$T_0$

$T_{pass}$

$GD_{pass}$

$f_{pass}$     freq

where

$$GD_{pass} = T_{pass}/T_0$$

$|H(f)|$

$A_{pass}$

$A_{stop}$

$f_{pass}$     $f_{stop}$

$|H(f)|$

$A_{pass}$

$BW_{pass}$

$BW_{stop}$

$A_{stop}$

$f_{center}$

$|H(f)|$

$A_{pass}$

$A_{stop}$

$f_{stop}$     $f_{pass}$

$|H(f)|$

$A_{pass}$

$BW_{pass}$

$BW_{stop}$

$A_{stop}$

$f_{center}$
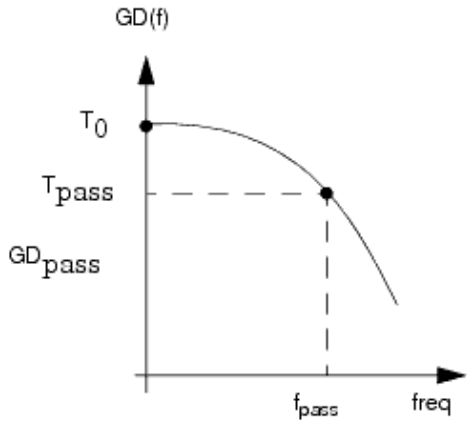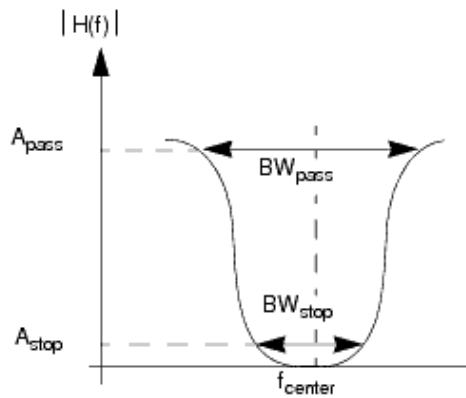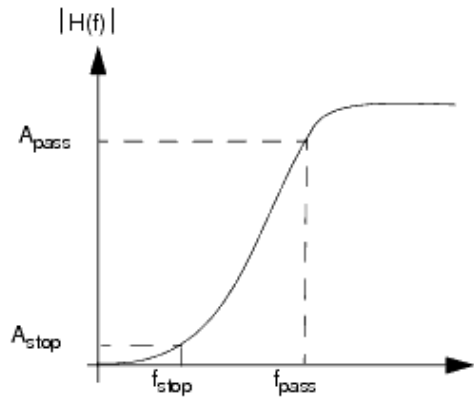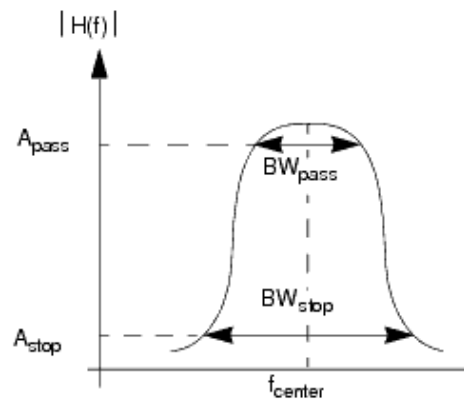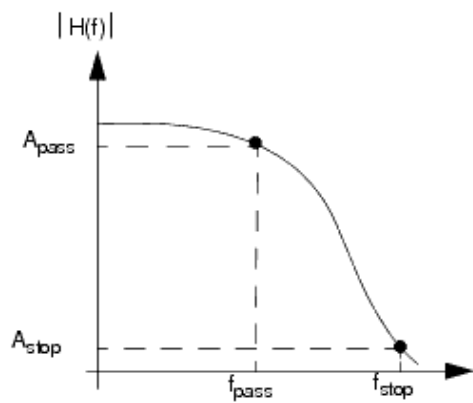
## Lowpass Filter Behavior at DC

At DC, a lowpass filter appears to reduce to a wire. During DC analysis, one would therefore intuitively expect the S-matrix of a lowpass filter to reduce to S11=S22=1 and S12=S21=0. Also, one would intuitively expect the input and output voltages and currents to fulfill v2=v1 and i2=i1. For Z1=Z2, both these expectations are met. However, if Z1 and Z2 are unequal, there is simply no way to mathematically realize a meaningful filter that meets both expectations. If it meets the expectations in the S-domain, it won't for voltages and currents and vice versa.

The ADS lowpass filters are implemented in the S-domain, not the voltage/current-domain. At DC, the lowpass filters therefore reduce to a perfect [1 0;0 1] S-matrix regardless of the choice of reference impedance. The output voltage and current, however, are given by v2=sqrt (Z2/Z1) × v1 and i2=sqrt(Z1/Z2) × i1. This, as pointed out previously, reduces to v2=v1 and i2=i1 only for Z1=Z2. These relations can easily be derived from power conservation and can also be found from more rigorous S-parameter analysis.

## References

1. A. I. Zverev, *Handbook of Filter Synthesis* , John Wiley and Sons, New York, 1967.
2. H. J. Blinchikoff and A. I. Zverev, *Filtering in the Time and Frequency Domains* , John Wiley and Sons, New York, 1976.
3. Kamilo Feher, *Digital Communications: Microwave Applications* , Prentice-Hall, Englewood Cliffs, N.J., 1981.
4. John G. Proakis, *Digital Communications* , McGraw-Hill, 1989.

## Components

- *BPF Bessel (Bandpass Filter, Bessel-Thompson)* (ccsys)
- *BPF Butterworth (Bandpass Filter, Butterworth)* (ccsys)
- *BPF Chebyshev (Bandpass Filter, Chebyshev)* (ccsys)
- *BPF Elliptic (Bandpass Filter, Elliptic)* (ccsys)
- *BPF Gaussian (Bandpass Filter, Gaussian)* (ccsys)
- *BPF PoleZero (Bandpass Filter, Pole Zero)* (ccsys)
- *BPF Polynomial (Bandpass Filter, Polynomial)* (ccsys)
- *BPF RaisedCos (Bandpass Filter, Raised-Cosine)* (ccsys)
- *BSF Bessel (Bandstop Filter, Bessel-Thompson)* (ccsys)
- *BSF Butterworth (Bandstop Filter, Butterworth)* (ccsys)
- *BSF Chebyshev (Bandstop Filter, Chebyshev)* (ccsys)
- *BSF Elliptic (Bandstop Filter, Elliptic)* (ccsys)
- *BSF Gaussian (Bandstop Filter, Gaussian)* (ccsys)
- *BSF PoleZero (Bandstop Filter, Pole Zero)* (ccsys)
- *BSF Polynomial (Bandstop Filter, Polynomial)* (ccsys)
- *BSF RaisedCos (Bandstop Filter, Raised-Cosine)* (ccsys)
- *HPF Bessel (Highpass Filter, Bessel-Thompson)* (ccsys)
- *HPF Butterworth (Highpass Filter, Butterworth)* (ccsys)
- *HPF Chebyshev (Highpass Filter, Chebyshev)* (ccsys)
- *HPF Elliptic (Highpass Filter, Elliptic)* (ccsys)
- *HPF Gaussian (Highpass Filter, Gaussian)* (ccsys)
- *HPF PoleZero (Highpass Filter, Pole Zero)* (ccsys)
- *HPF Polynomial (Highpass Filter, Polynomial)* (ccsys)
- *HPF RaisedCos (Highpass Filter, Raised-Cosine)* (ccsys)
- *LPF Bessel (Lowpass Filter, Bessel-Thompson)* (ccsys)
- *LPF Butterworth (Lowpass Filter, Butterworth* (ccsys)
- *LPF Chebyshev (Lowpass Filter, Chebyshev)* (ccsys)
- *LPF Elliptic (Lowpass Filter, Elliptic)* (ccsys)
- *LPF Gaussian (Lowpass Filter, Gaussian)* (ccsys)
- *LPF GMSK (Lowpass Filter, GMSK)* (ccsys)
- *LPF PoleZero (Lowpass Filter, Pole Zero)* (ccsys)

- *LPF Polynomial (Lowpass Filter, Polynomial)* (ccsys)
- *LPF RaisedCos (Lowpass Filter, Raised-Cosine)* (ccsys)
- *SAW Filter (Saw Filter)* (ccsys)

# BPF_Bessel (Bandpass Filter, Bessel-Thompson)

## Symbol



## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Fcenter | Passband center frequency | GHz | 1.5 |
| BWpass | Passband edge-to-edge width | GHz | 1.0 |
| Apass | Attenuation at stopband edges | dB | 3 |
| GDpass | Group delay rolloff at passband edge, 0 < GDpass < 1 | None | 0.9 |
| StopType | Stopband input impedance type: OPEN or SHORT | None | open |
| MaxRej | Maximum rejection level | dB | None |
| N | Filter Order (if N > 0, it overwrites GDpass) | None | 0 |
| IL | Passband insertion loss | dB | 0 |
| Qu | Unloaded quality factor for resonators, default setting is an infinite Qu and expresses a dissipationless resonant circuit. | None | 1e308 |
| Z1 | Input port reference impedance | Ohm | 50 |
| Z2 | Output port reference impedance | Ohm | 50 |
| Temp | Temperature | °C | None |

## Range of Usage

BWpass < Fcenter
$0.01 \leq$ Apass $\leq 3.0$
$0 <$ GDpass $< 1$
$1 \leq N \leq 15$
$Qu \geq 1$

## Notes/Equations

1. Refer to *Filter Categories* (ccsys).
2. This component has no default artwork associated with it.

# BPF_Butterworth (Bandpass Filter, Butterworth)

## Symbol



## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Fcenter | Passband center frequency | GHz | 1.5 |
| BWpass | Passband edge-to-edge width | GHz | 1.0 |
| Apass | Attenuation at stopband edges | dB | 3 |
| BWstop | Stopband edge-to-edge width | GHz | 1.2 |
| Astop | Attenuation at stopband edges | dB | 20 |
| StopType | Stopband input impedance type: OPEN or SHORT | None | open |
| MaxRej | Maximum rejection level | dB | None |
| N | Filter Order (if N > 0, it overwrites GDpass) | None | 0 |
| IL | Passband insertion loss | dB | 0 |
| Qu | Unloaded quality factor for resonators, default setting is an infinite Qu and expresses a dissipationless resonant circuit. | None | 1e308 |
| Z1 | Input port reference impedance | Ohm | 50 |
| Z2 | Output port reference impedance | Ohm | 50 |
| Temp | Temperature | °C | None |

## Range of Usage

$1 \leq N \leq 15$

BWpass < Fcenter

$0.01 \leq Apass \leq 3.0$

$Qu \geq 1$

## Notes/Equations

1. Refer to *Filter Categories* (ccsys).
2. This component has no default artwork associated with it.

# BPF_Chebyshev (Bandpass Filter, Chebyshev)

## Symbol



## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Fcenter | Passband center frequency | GHz | 1.5 |
| BWpass | Passband edge-to-edge width | GHz | 1.0 |
| Apass | Attenuation at stopband edges | dB | 1 |
| Ripple | Passband ripple | dB | 1 |
| BWstop | width measured from lower to upper stopband edges | GHz | 1.2 |
| Astop | attenuation at stopband edges | dB | 20 |
| StopType | Stopband input impedance type: OPEN or SHORT | None | open |
| MaxRej | Maximum rejection level | dB | None |
| N | Filter Order (if N > 0, it overwrites GDpass) | None | 0 |
| IL | Passband insertion loss | dB | 0 |
| Qu | Unloaded quality factor for resonators, default setting is an infinite Qu and expresses a dissipationless resonant circuit. | None | 1e308 |
| Z1 | Input port reference impedance | Ohm | 50 |
| Z2 | Output port reference impedance | Ohm | 50 |
| Temp | Temperature | °C | None |

## Range of Usage

BWpass < Fcenter
$0.01 \leq Ripple \leq 3.0$
$1 \leq N \leq 15$
$Qu \geq 1$

## Notes/Equations

1. Refer to *Filter Categories* (ccsys).
2. This component has no default artwork associated with it.

# BPF_Elliptic (Bandpass Filter, Elliptic)
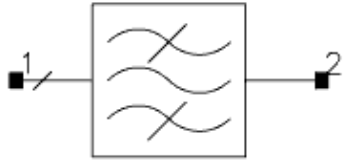
## Symbol



## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Fcenter | Passband center frequency | GHz | 1.5 |
| BWpass | Passband edge-to-edge width | GHz | 1.0 |
| Ripple | Passband ripple | dB | 1 |
| BWstop | Stopband edge-to-edge width | GHz | 1.2 |
| Astop | Attenuation at stopband edges | dB | 20 |
| StopType | Stopband input impedance type: OPEN or SHORT | None | open |
| MaxRej | Maximum rejection level | dB | None |
| N | Filter Order (if N > 0, it overwrites GDpass) | None | 0 |
| IL | Passband insertion loss | dB | 0 |
| Qu | Unloaded quality factor for resonators, default setting is an infinite Qu and expresses a dissipationless resonant circuit. | None | 1e308 |
| Z1 | Input port reference impedance | Ohm | 50 |
| Z2 | Output port reference impedance | Ohm | 50 |
| Temp | Temperature | °C | None |

## Range of Usage

BWpass < Fcenter
$0.01 \leq Ripple \leq 3.0$
Astop > 0
$1 \leq N \leq 15$

## Notes/Equations

1. Refer to *Filter Categories* (ccsys).
2. This component has no default artwork associated with it.

# BPF_Gaussian (Bandpass Filter, Gaussian)

## Symbol



## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Fcenter | Passband center frequency | GHz | 1.5 |
| BWpass | Passband edge-to-edge width | GHz | 1.0 |
| Apass | Attenuation at stopband edges | dB | 3 |
| GDpass | Group delay rolloff at passband edge, 0 < GDpass < 1 | None | 0.9 |
| StopType | Stopband input impedance type: OPEN or SHORT | None | open |
| MaxRej | Maximum rejection level | dB | None |
| N | Filter Order (if N > 0, it overwrites GDpass) | None | 0 |
| IL | Passband insertion loss | dB | 0 |
| Qu | Unloaded quality factor for resonators, default setting is an infinite Qu and expresses a dissipationless resonant circuit | None | 1e308 |
| Z1 | Input port reference impedance | Ohm | 50 |
| Z2 | Output port reference impedance | Ohm | 50 |
| Temp | Temperature | °C | None |

## Range of Usage

BWpass < Fcenter
$0.01 \leq Apass \leq 3.0$
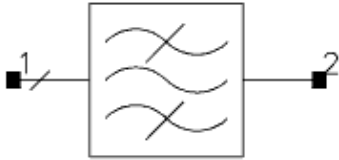$0 < GDpass < 1$
$1 \leq N \leq 15$
$Qu \geq 1$

## Notes/Equations

1. Refer to *Filter Categories* (ccsys).
2. This component has no default artwork associated with it.

# BPF_PoleZero (Bandpass Filter, Pole Zero)

## Symbol



## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Numerator | List of numerator coefficients | None | list(1) |
| Denominator | List of denominator coefficients | None | list(1,1.4,1) |
| Gain | Gain factor | None | 1.0 |
| Fcenter | Passband edge frequency | GHz | 1 |
| BWpass | Passband edge-to-edge width | GHz | 1.0 |
| StopType | Stopband input impedance type: OPEN or SHORT | None | open |
| Z1 | Input port reference impedance | Ohm | 50 |
| Z2 | Output port reference impedance | Ohm | 50 |

## Notes/Equations

1. This is an S-domain filter.
2. Denominator and Numerator are a list of polynomial coefficients.
   The transfer function for the filter is:

$$S_{21} = Gain\left(\frac{N_0 + (N_1 \times S) + (N_2 \times S^2)...}{D_0 + (D_1 \times S) + (D_2 \times S^2)...}\right)$$

   where
   $S = j \times (Freq/F_o - F_o/Freq)/(F_{high}/F_o - F_o/F_{high})$

   and
   Freq is the analysis frequency
   Fhigh = Fcenter + 0.5 × BWpass
   Fo = sqrt((Fcenter − 0.5 × BWpass) × (Fcenter + 0.5 × BWpass))

3. The following example demonstrates interpretation of simulation results with this component. From the user-specified poles/zeros, we derive:
   *S21_Lowpass_Prototype= Gain*[(s-Z1)*...*(s-Zn)]/[(s-P1)*...(s-Pm)]*
   We then check to see if *S21_Lowpass_Protope* is > 1. If yes, we scale S21 by a another factor to make sure *S21_Max ≤ 1* . We then derive S11 (S22) from the following formula:

   *S11$^2$ + S21$^2$ = 1*

   In this example, when Gain is set > 0.471151, then S11 is derived as what you will expect. If Gain in your example is < 0.471151, then S11 derived from the preceding

equation, will be much higher that what you will expect. In this situation, set Gain to be 0.1 so that S21 has a lot of insertion loss. But we assumed there is no insertion loss in deriving S11.

There are other alternatives:

- Use S2P_Eqn so that you can define the S21 and S11 polynomials however you want. You can define this as follows:

  *s=j ω , S21=Gain\*(s-Z1)\*...\*(s-Zn)/(s-P1)\*...\*(s-Pn), S11=<your choice>*

- Use BPF_Pole_Zero to model a lossless BPF, then use an attenuator to add insertion loss.

# BPF_Polynomial (Bandpass Filter, Polynomial)

## Symbol



## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Numerator | List of numerator coefficients | None | list(1) |
| Denominator | List of denominator coefficients | None | list(1,1.4,1) |
| Gain | Gain factor | None | 1.0 |
| Fcenter | Passband center frequency | GHz | 1 |
| BWpass | Passband edge-to-edge width | GHz | 1.0 |
| StopType | Stopband input impedance type: OPEN or SHORT | None | open |
| Z1 | Input port reference impedance | Ohm | 50 |
| Z2 | Output port reference impedance | Ohm | 50 |

## Notes/Equations

1. This is an S-domain filter.
2. Denominator and Numerator are a list of polynomial coefficients. The transfer function for the filter is:

$$S_{21} = Gain\left(\frac{N_0 + (N_1 \times S) + (N_2 \times S^2)...}{D_0 + (D_1 \times S) + (D_2 \times S^2)...}\right)$$

where
$S = j \times (Freq/F_o - F_o/Freq)/(F_{high}/F_o - F_o/F_{high})$

and
Freq is the analysis frequency
$F_{high} = Fcenter + 0.5 \times BWpass$
$F_o = sqrt((Fcenter - 0.5 \times BWpass) \times (Fcenter + 0.5 \times BWpass))$

121

# BPF_RaisedCos (Bandpass Filter, Raised-Cosine)

## Symbol



## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Alpha | Rolloff factor defining filter's excess bandwidth, 0 ≤ Alpha ≤ 1 | None | 0.35 |
| Fcenter | Passband center frequency | GHz | 1.5 |
| SymbolRate | Digital symbol rate defining filter bandwidth | kHz | 24.3 |
| DelaySymbols | Number of symbols delayed by filter | None | 5 |
| Exponent | Exponent Factor ( 0 ≤ Exponent ≤ 1 ), to provide for Root Raised-Cosine filter | None | 0.5 |
| DutyCycle | Pulse duty cycle in percent, used for sinc(x) correction | None | 0 |
| SincE | Flag to include the Exponent factor on the sinc(x) correction: yes or no | None | no |
| Gain | Gain factor | None | 1.0 |
| Zout | Output impedance | Ohm | 50 |
| WindowType | Window type applied to impulse response. 0=None, 1=Hann, 2=Hamming | None | 0 |
| ImpMaxFreq | Maximum frequency to consider when calculating impulse response | | |
| ImpDeltaFreq | Impulse sample spacing in frequency | | |
| ImpMaxPts | Maximum number of points in impulse response | None | None |
| Other | Output string to netlist | None | None |

## Range of Usage

0 ≤ Alpha ≤ 1
DelaySymbols ≥ 1
0 ≤ Exponent ≤ 1
0 ≤ DutyCycle ≤ 100

## Notes/Equations

1. Refer to *Filter Categories* (ccsys).
2. This filter is unidirectional; input impedance is infinite; output impedance is specified by Zout.
3. Voltage gain is described by the following function.

$$\frac{V_{out}}{V_{in}} = \frac{Gain \times G_{filt}}{G_{comp}} e^{-j2\pi frequency\left(\frac{DelaySymbols}{SymbolRate}\right)}$$

where

| | | |
|---|---|---|
| $G_{filt}$ | = 1.0 for *Freq* ≤ 0.5 × (1 − *Alpha* ) × *SymbolRate* | |
| | = 0.0 for *Freq* ≥ 0.5 × (1 + *Alpha* ) × *SymbolRate* | |
| | = [0.5 × (1 − sin[п × ( *Freq* − *SymbolRate* /2)/ | |
| $G_{comp}$ | = 1.0 | if *DutyCycle* =0 |
| | = [0.01 × *DutyCycle* × sinc(x)]$^{Exponent}$ | if *SincE* = YES |
| | = [0.01 × *DutyCycle* × sinc(x)] | if *SincE* = NO |
| *Freq* | = abs ( *Fcenter* − *frequency* ) | |
| sinc(x) | = sin(x)/x | |
| x | = 0.01 × *DutyCycle* × п × *Freq* / *SymbolRate* | |

4. While Exponent can be any value, the standard value is 1.0 for the ideal raised-cosine filter response or 0.5 to simulate the root raised-cosine filter response when present at both the receiving and transmitting channels.

5. In steady-state frequency-domain analyses, the ideal frequency-domain response described previously is used; however, this ideal response has an infinite duration impulse response that must be approximated for time-domain simulations in either Transient or Circuit Envelope. If DelaySymbols is set too small, then the impulse response will be severely truncated and will not accurately reflect the ideal frequency response.
   Using a Hanning window (WindowType=1) a DelaySymbols parameter of 15 should result in equivalent frequency domain sidelobes of −90dBc or smaller.

6. Accuracy of this model in Transient or Circuit Envelope can be further controlled through the ImpMaxFreq, ImpDeltaFreq, and ImpMaxPts parameters.

7. The filter can include gain equalization to compensate for duty cycle roll-off. If DutyCycle = 0.0, then no compensation will be applied. If SincE=YES, Exponent will be applied to the gain compensation term $G_{comp}$. The Exponent term is always present in the $G_{filt}$ term.

8. This component has no default artwork associated with it.

# BSF_Bessel (Bandstop Filter, Bessel-Thompson)

## Symbol



## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Fcenter | Stopband center frequency | GHz | 1.5 |
| BWpass | Passband edge-to-edge width | GHz | 1.0 |
| Apass | Attenuation at passband edges | dB | 3 |
| GDpass | Group delay rolloff at passband edge, 0 < GDpass < 1 | None | 0.9 |
| StopType | Stopband input impedance type: OPEN or SHORT | None | open |
| MaxRej | Maximum rejection level | dB | None |
| N | Filter order (if N > 0, it overwrites GDpass); if not given, it is calculated based on BWpass, Apass and GDpass | None | 0 |
| IL | Passband insertion loss | dB | 0 |
| Qu | Unloaded quality factor for resonators, default setting is an infinite Qu and expresses a dissipationless resonant circuit. | None | 1e308 |
| Z1 | Input port reference impedance | Ohm | 50 |
| Z2 | Output port reference impedance | Ohm | 50 |
| Temp | Temperature | °C | None |

## Range of Usage

BWpass < Fcenter
$0.01 \leq Apass \leq 3.0$
$0 < GDpass < 1$
$1 \leq N \leq 15$

## Notes/Equations

1. Refer to *Filter Categories* (ccsys).
2. This component has no default artwork associated with it.

# BSF_Butterworth (Bandstop Filter, Butterworth)

## Symbol



## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Fcenter | Stopband center frequency | GHz | 1.5 |
| BWstop | Stopband edge-to-edge width | GHz | 1.0 |
| Astop | Attenuation at stopband edges | dB | 20 |
| BWpass | Passband edge-to-edge width | GHz | 1.2 |
| Apass | Attenuation at passband edges | dB | 3 |
| StopType | Stopband input impedance type: OPEN or SHORT | None | open |
| MaxRej | Maximum rejection level | dB | None |
| N | Filter order (if N > 0, it overwrites BWstop and Astop)); if not given, it is calculated based on BWpass, Apass and GDpass | None | 0 |
| IL | Passband insertion loss | dB | 0 |
| Qu | Unloaded quality factor for resonators, default setting is an infinite Qu and expresses a dissipationless resonant circuit. | None | 1e308 |
| Z1 | Input port reference impedance | Ohm | 50 |
| Z2 | Output port reference impedance | Ohm | 50 |
| Temp | Temperature | °C | None |

## Range of Usage

BWpass < Fcenter
$0.01 \leq Apass \leq 3.0$
$1 \leq N \leq 15$

## Notes/Equations

1. Refer to *Filter Categories* (ccsys).
2. This component has no default artwork associated with it.

# BSF_Chebyshev (Bandstop Filter, Chebyshev)

## Symbol



## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Fcenter | Stopband center frequency | GHz | 1.5 |
| BWstop | Stopband edge-to-edge width | GHz | 1.0 |
| Astop | Attenuation at stopband edges | dB | 20 |
| Ripple | Stopband ripple | dB | 1 |
| BWpass | Passband edge-to-edge width | GHz | 1.2 |
| Apass | Attenuation at passband edges | dB | 1 |
| StopType | Stopband input impedance type: OPEN or SHORT | None | open |
| MaxRej | Maximum rejection level | dB | None |
| N | Filter order (if N > 0, it overwrites BWstop and Astop)); if not given, it is calculated based on BWpass, Apass and GDpass | None | 0 |
| IL | Passband insertion loss | dB | 0 |
| Qu | Unloaded quality factor for resonators, default setting is an infinite Qu and expresses a dissipationless resonant circuit. | None | 1e308 |
| Z1 | Input port reference impedance | Ohm | 50 |
| Z2 | Output port reference impedance | Ohm | 50 |
| Temp | Temperature | °C | None |

## Range of Usage

BWpass < Fcenter
$0.01 \leq Ripple \leq 3.0$
$1 \leq N \leq 15$

## Notes/Equations

1. Refer to *Filter Categories* (ccsys).
2. This component has no default artwork associated with it.

# BSF_Elliptic (Bandstop Filter, Elliptic)

## Symbol



## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Fcenter | Stopband center frequency | GHz | 1.5 |
| BWstop | Stopband edge-to-edge width | GHz | 1.0 |
| Astop | Attenuation at stopband edges | dB | 20 |
| Ripple | Stopband ripple | dB | 1 |
| BWpass | Passband edge-to-edge width | GHz | 1.2 |
| Apass | Attenuation at passband edges | dB | 1 |
| StopType | Stopband input impedance type: OPEN or SHORT | None | open |
| MaxRej | Maximum rejection level | dB | None |
| N | Filter order (if N > 0, it overwrites BWstop and Astop)); if not given, it is calculated based on BWpass, Apass and GDpass | None | 0 |
| IL | Passband insertion loss | dB | 0 |
| Qu | Unloaded quality factor for resonators, default setting is an infinite Qu and expresses a dissipationless resonant circuit. | None | 1e308 |
| Z1 | Input port reference impedance | Ohm | 50 |
| Z2 | Output port reference impedance | Ohm | 50 |
| Temp | Temperature | °C | None |

## Range of Usage

BWpass < Fcenter
$0.01 \leq Ripple \leq 3.0$
$1 \leq N \leq 15$

## Notes/Equations

1. Refer to *Filter Categories* (ccsys).
2. This component has no default artwork associated with it.

# BSF_Gaussian (Bandstop Filter, Gaussian)

## Symbol



## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Fcenter | Stopband center frequency | GHz | 1.5 |
| BWpass | Passband edge-to-edge width | GHz | 1.0 |
| Apass | Attenuation at passband edges | dB | 3 |
| GDpass | Group delay rolloff at passband edge, 0 < GDpass < 1 | None | 0.9 |
| StopType | Stopband input impedance type: OPEN or SHORT | None | open |
| MaxRej | Maximum rejection level | dB | None |
| N | Filter order (if N > 0, it overwrites BWstop and Astop)); if not given, it is calculated based on BWpass, Apass and GDpass | None | 0 |
| IL | Passband insertion loss | dB | 0 |
| Qu | Unloaded quality factor for resonators, default setting is an infinite Qu and expresses a dissipationless resonant circuit. | None | 1e308 |
| Z1 | Input port reference impedance | Ohm | 50 |
| Z2 | Output port reference impedance | Ohm | 50 |
| Temp | Temperature | °C | None |

## Range of Usage

BWpass < Fcenter
$0.01 \leq Apass \leq 3.0$
$0 < GDpass < 1$
$1 \leq N \leq 15$

## Notes/Equations
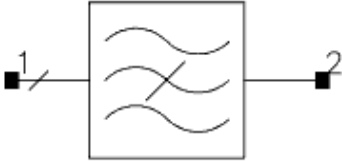
1. Refer to *Filter Categories* (ccsys).
2. This component has no default artwork associated with it.

# BSF_PoleZero (Bandstop Filter, Pole Zero)

## Symbol



## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Numerator | List of numerator coefficients | None | list(1) |
| Denominator | List of denominator coefficients | None | list(1,1.4,1) |
| Gain | Gain factor | None | 1.0 |
| Fcenter | Passband edge frequency | GHz | 1 |
| BWpass | Passband edge-to-edge width | GHz | 1.0 |
| StopType | Stopband input impedance type: OPEN or SHORT | None | open |
| Z1 | Input port reference impedance | Ohm | 50 |
| Z2 | Output port reference impedance | Ohm | 50 |

## Notes/Equations

1. This is an S-domain filter.
2. Denominator and Numerator are lists of polynomial coefficients. The transfer function for the filter is:

$$S_{21} = Gain\left(\frac{N_0 + (N_1 \times S) + (N_2 \times S^2)...}{D_0 + (D_1 \times S) + (D_2 \times S^2)...}\right)$$

where
$S = -j \times (F_o/F_{low} - F_{low}/F_o)/(Freq/F_o - F_o/Freq)$

and
Freq is the analysis frequency
$F_{low}$ = Fcenter − 0.5 × BWpass

$F_o$ = sqrt((Fcenter - 0.5 × BWpass) × Fcenter + 0.5 × BWpass))

At least one Denominator coefficient must be supplied.

# BSF_Polynomial (Bandstop Filter, Polynomial)

**Symbol**



**Parameters**

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Numerator | List of numerator coefficients | None | list(1) |
| Denominator | List of denominator coefficients | None | list(1,1.4,1) |
| Gain | Gain factor | None | 1.0 |
| Fcenter | Passband Edge Frequency | GHz | 1 |
| BWpass | Passband Edge-to-Edge Width | GHz | 1.0 |
| StopType | Stopband input impedance type: OPEN or SHORT | None | open |
| Z1 | Input port reference impedance | Ohm | 50 |
| Z2 | Output port reference impedance | Ohm | 50 |

**Notes/Equations**

1. This is an S-domain filter.
2. Denominator and Numerator are lists of polynomial coefficients. The transfer function for the filter is:

$$S_{21} = Gain\left(\frac{N_0 + (N_1 \times S) + (N_2 \times S^2)...}{D_0 + (D_1 \times S) + (D_2 \times S^2)...}\right)$$

where

$S = -j \times (F_o/F_{low} - F_{low}/F_o)/(Freq/F_o - F_o/Freq)$

and
Freq is the analysis frequency
$F_{low} = Fcenter - 0.5 \times BWpass$

$F_o = sqrt((Fcenter - 0.5 \times BWpass) \times Fcenter + 0.5 \times BWpass))$

At least one Denominator coefficient must be supplied.

# BSF_RaisedCos (Bandstop Filter, Raised-Cosine)

## Symbol



## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Alpha | Rolloff factor defining filter excess bandwidth, 0 ≤ Alpha ≤ 1 | None | 0.35 |
| Fcenter | Passband center frequency | GHz | 1.5 |
| SymbolRate | Digital symbol rate defining filter bandwidth | kHz | 24.3 |
| DelaySymbols | Number of symbols delayed by filter | None | 5 |
| Exponent | Exponent factor ( 0 ≤ Exponent ≤ 1 ), to provide for Root Raised-Cosine filter | None | 0.5 |
| DutyCycle | Pulse duty cycle in percent, used for sinc(x) correction | None | 0 |
| SincE | Flag to include the Exponent factor on the sinc(x) correction: yes or no | None | no |
| Gain | Gain factor | None | 1.0 |
| Zout | Output impedance | Ohm | 50 |
| WindowType | Window type applied to impulse response: 0=None, 1=Hann, 2=Hamming | None | 0 |
| ImpMaxFreq | Maximum frequency to consider when calculating impulse response | | |
| ImpDeltaFreq | Frequency sample spacing when calculating impulse response | | |
| ImpMaxPts | Maximum number of points in impulse response | None | None |
| Other | Output string to netlist | None | None |

## Range of Usage

0 ≤ Alpha ≤ 1
DelaySymbols ≥ 1
0 ≤ Exponent ≤ 1
0 ≤ DutyCycle ≤ 100

## Notes/Equations

1. Refer to *Filter Categories* (ccsys).
2. This filter is unidirectional; input impedance is infinite; output impedance is specified by Zout.
3. Voltage gain is described by the following function.

$$\frac{V_{out}}{V_{in}} = \frac{Gain \times G_{filt}}{G_{comp}} e^{-j2\pi frequency\left(\frac{DelaySymbols}{SymbolRate}\right)}$$

| $G_{filt}$ | = 0.0 for *Freq* ≤ 0.5 × (1 − *Alpha* ) × *SymbolRate* | |
|---|---|---|
| | = 1.0 for *Freq* ≥ 0.5 × (1 + *Alpha* ) × *SymbolRate* | |
| | = [0.5 × (1 + sin[п × ( *Freq* − *SymbolRate* /2)/ | |
| $G_{comp}$ | = 1.0 | if *DutyCycle* =0 |
| | = [0.01 × *DutyCycle*  × sinc(x)]$^{Exponent}$ | if *SincE* = YES |
| | = [0.01 × *DutyCycle*  × sinc(x)] | if *SincE* = NO |
| *Freq* | = abs ( *Fcenter* − *frequency* ) | |
| sinc(x) | = sin(x)/x | |
| x | = 0.01 × *DutyCycle* × п × *Freq* / *SymbolRate* | |

4. While Exponent can be any value, the standard value is 1.0 for the ideal raised-cosine filter response or 0.5 to simulate the root raised-cosine filter response when present at both the receiving and transmitting channels.
5. In steady-state frequency-domain analyses, the ideal frequency-domain response described previously is used; however, this ideal response has an infinite duration impulse response that must be approximated for time-domain simulations in either Transient or Circuit Envelope. If DelaySymbols is set too small, then the impulse response will be severely truncated and will not accurately reflect the ideal frequency response.
   Using a Hanning window (WindowType=1) a DelaySymbols value of 15 should result in equivalent frequency domain sidelobes of −90dBc or smaller.
6. Accuracy of this model in Transient or Circuit Envelope can be further controlled through the ImpMaxFreq, ImpDeltaFreq, and ImpMaxPts parameters.
7. The filter can include gain equalization to compensate for duty cycle roll-off. If DutyCycle = 0.0, then no compensation will be applied. If SincE=YES, Exponent will be applied to the gain compensation term, $G_{comp}$. The Exponent term is always present in the $G_{filt}$ term.
8. This component has no default artwork associated with it.

# HPF_Bessel (Highpass Filter, Bessel-Thompson)

## Symbol



## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Fpass | Passband edge frequency | GHz | 1 |
| Apass | Attenuation at passband edges | dB | 3 |
| GDpass | Group delay rolloff at passband edge, 0 < GDpass < 1 | None | 0.9 |
| StopType | Stopband input impedance type: OPEN or SHORT | None | open |
| MaxRej | Maximum rejection level | dB | None |
| N | Filter order (if N > 0, it overwrites BWstop and Astop)); if not given, it is calculated based on BWpass, Apass and GDpass | None | 0 |
| IL | Passband insertion loss | dB | 0 |
| Qu | Unloaded quality factor for resonators, default setting is an infinite Qu and expresses a dissipationless resonant circuit. | None | 1e308 |
| Z1 | Input port reference impedance | Ohm | 50 |
| Z2 | Output port reference impedance | Ohm | 50 |
| Temp | Temperature | °C | None |

## Range of Usage

Fpass > 0
$0.01 \leq Apass \leq 3.0$
$0 \leq GDpass < 1$
$1 \leq N \leq 15$

## Notes/Equations

1. Refer to *Filter Categories* (ccsys).
2. This component has no default artwork associated with it.

# HPF_Butterworth (Highpass Filter, Butterworth)

## Symbol



## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Fpass | Passband edge frequency | GHz | 1 |
| Apass | Attenuation at passband edges | dB | 3 |
| Fstop | Stopband edge frequency | GHz | 0.8 |
| Astop | Attenuation at stopband edges | dB | 20 |
| StopType | Stopband input impedance type: OPEN or SHORT | None | open |
| MaxRej | Maximum rejection level | dB | None |
| N | Filter order (if N > 0, it overwrites BWstop and Astop)); if not given, it is calculated based on BWpass, Apass and GDpass | None | 0 |
| IL | Passband insertion loss | dB | 0 |
| Qu | Unloaded quality factor for resonators, default setting is an infinite Qu and expresses a dissipationless resonant circuit. | None | 1e308 |
| Z1 | Input port reference impedance | Ohm | 50 |
| Z2 | Output port reference impedance | Ohm | 50 |
| Temp | Temperature | °C | None |

## Range of Usage

Fpass > 0
$0.01 \leq Apass \leq 3.0$
$1 \leq N \leq 15$

## Notes/Equations

1. Refer to *Filter Categories* (ccsys).
2. This component has no default artwork associated with it.

# HPF_Chebyshev (Highpass Filter, Chebyshev)

## Symbol



## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Fpass | Passband edge frequency | GHz | 1 |
| Apass | Attenuation at passband edges | dB | 1 |
| Ripple | Stopband ripple | dB | 1 |
| Fstop | Stopband edge frequency | GHz | 0.8 |
| Astop | Attenuation at stopband edges | dB | 20 |
| StopType | Stopband input impedance type: OPEN or SHORT | None | open |
| MaxRej | Maximum rejection level | dB | None |
| N | Filter order (if N > 0, it overwrites BWstop and Astop)); if not given, it is calculated based on BWpass, Apass and GDpass | None | 0 |
| IL | Passband insertion loss | dB | 0 |
| Qu | Unloaded quality factor for resonators, default setting is an infinite Qu and expresses a dissipationless resonant circuit. | None | 1e308 |
| Z1 | Input port reference impedance | Ohm | 50 |
| Z2 | Output port reference impedance | Ohm | 50 |
| Temp | Temperature | °C | None |

## Range of Usage

Fpass > 0
0.01 ≤ Ripple ≤ 3.0
1 ≤ N ≤ 15

## Notes/Equations

1. Refer to *Filter Categories* (ccsys).
2. This component has no default artwork associated with it.

# HPF_Elliptic (Highpass Filter, Elliptic)

## Symbol

## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Fpass | Passband edge frequency | GHz | 1 |
| Ripple | Stopband ripple | dB | 1 |
| Fstop | Stopband edge frequency | GHz | 0.8 |
| Astop | Attenuation at stopband edges | dB | 20 |
| StopType | Stopband input impedance type: OPEN or SHORT | None | open |
| MaxRej | Maximum rejection level | dB | None |
| N | Filter order (if N > 0, it overwrites BWstop and Astop)); if not given, it is calculated based on BWpass, Apass and GDpass | None | 0 |
| IL | Passband insertion loss | dB | 0 |
| Qu | Unloaded quality factor for resonators, default setting is an infinite Qu and expresses a dissipationless resonant circuit. | None | 1e308 |
| Z1 | Input port reference impedance | Ohm | 50 |
| Z2 | Output port reference impedance | Ohm | 50 |
| Temp | Temperature | °C | None |

## Range of Usage

Fpass > 0
0.01 ≤ Ripple ≤ 3.0
Astop > 0
1 ≤ N ≤ 15

## Notes/Equations

1.  Refer to *Filter Categories* (ccsys).
2.  This component has no default artwork associated with it.

# HPF_Gaussian (Highpass Filter, Gaussian)

## Symbol



## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Fpass | Passband edge frequency | GHz | 1 |
| Apass | Attenuation at passband edges | dB | 3 |
| GDpass | Group delay rolloff at passband edge, 0 < GDpass < 1 | None | 0.9 |
| StopType | Stopband input impedance type: OPEN or SHORT | None | open |
| MaxRej | Maximum rejection level | dB | None |
| N | Filter order (if N > 0, it overwrites BWstop and Astop)); if not given, it is calculated based on BWpass, Apass and GDpass | None | 0 |
| IL | Passband insertion loss | dB | 0 |
| Qu | Unloaded quality factor for resonators, default setting is an infinite Qu and expresses a dissipationless resonant circuit. | None | 1e308 |
| Z1 | Input port reference impedance | Ohm | 50 |
| Z2 | Output port reference impedance | Ohm | 50 |
| Temp | Temperature | °C | None |

## Range of Usage

Fpass > 0
0.01 ≤ Apass ≤ 3.0
0 < GDpass < 1
1 ≤ N ≤ 15

## Notes/Equations

1. Refer to *Filter Categories* (ccsys).
2. This component has no default artwork associated with it.

# HPF_PoleZero (Highpass Filter, Pole Zero)

## Symbol



## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Numerator | List of numerator coefficients | None | list(1) |
| Denominator | List of denominator coefficients | None | list(1,1.4,1) |
| Gain | Gain factor | None | 1.0 |
| Fpass | Passband Edge Frequency | GHz | 1 |
| StopType | Stopband input impedance type: OPEN or SHORT | None | open |
| Z1 | Input port reference impedance | Ohm | 50 |
| Z2 | Output port reference impedance | Ohm | 50 |

## Notes/Equations

1. This is an S-domain filter.
2. Denominator and Numerator are a list of polynomial coefficients. The transfer function for the filter is:

$$S_{21} = Gain\left(\frac{N_0 + (N_1 \times S) + (N_2 \times S^2)...}{D_0 + (D_1 \times S) + (D_2 \times S^2)...}\right)$$

where
S = −j(Fpass/Freq)
and
Freq is the analysis frequency

# HPF_Polynomial (Highpass Filter, Polynomial)

## Symbol



## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Numerator | List of numerator coefficients | None | list(1) |
| Denominator | List of denominator coefficients | None | list(1,1.4,1) |
| Gain | Gain factor | None | 1.0 |
| Fpass | Passband edge frequency | GHz | 1 |
| StopType | Stopband input impedance type: OPEN or SHORT | None | open |
| Z1 | Input port reference impedance | Ohm | 50 |
| Z2 | Output port reference impedance | Ohm | 50 |

## Notes/Equations

1. This is an S-domain filter.
2. Denominator and Numerator are a list of polynomial coefficients.
   The transfer function for the filter is:

$$S_{21} = Gain\left(\frac{N_0 + (N_1 \times S) + (N_2 \times S^2)...}{D_0 + (D_1 \times S) + (D_2 \times S^2)...}\right)$$

   where
   S = −j(Fpass/Freq)
   and
   Freq is the analysis frequency

# HPF_RaisedCos (Highpass Filter, Raised-Cosine)

## Symbol



## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Alpha | Rolloff factor defining filter excess bandwidth, 0 ≤ Alpha ≤ 1 | None | 0.35 |
| SymbolRate | Digital symbol rate defining filter bandwidth | kHz | 24.3 |
| DelaySymbols | Number of symbols delayed by filter | None | 5 |
| Exponent | Exponent factor ( 0 ≤ Exponent ≤ 1 ), to provide for Root Raised-Cosine filter | None | 0.5 |
| DutyCycle | Pulse duty cycle in percent, used for sinc(x) correction | None | 0 |
| SincE | Flag to include the Exponent factor on the sinc(x) correction: yes or no | None | no |
| Gain | Gain factor | None | 1.0 |
| Zout | Output impedance | Ohm | 50 |
| WindowType | Window type applied to impulse response: 0=None, 1=Hann, 2=Hamming | None | 0 |
| ImpMaxFreq | Maximum frequency to consider when calculating impulse response | | |
| ImpDeltaFreq | Frequency sample spacing when calculating impulse response | | |
| ImpMaxPts | Maximum number of points in impulse response | None | None |
| Other | Output string to netlist | None | None |

## Range of Usage

0 ≤ Alpha ≤ 1
DelaySymbols ≥ 1
0 ≤ Exponent ≤ 1
0 ≤ DutyCycle ≤ 100

## Notes/Equations

1. Refer to *Filter Categories* (ccsys).
2. This filter is unidirectional; input impedance is infinite; output impedance is specified by Zout.
3. This filter is unidirectional; input impedance is infinite; output impedance is specified by Zout.
4. Voltage gain is described by the following function.

$$\frac{V_{out}}{V_{in}} = \frac{Gain \times G_{filt}}{G_{comp}} e^{-j2\pi frequency\left(\frac{Delay\,Symbols}{Symbol\,Rate}\right)}$$

where:

| $G_{filt}$ | = 0.0 for $frequency \leq 0.5 \times (1 - Alpha) \times SymbolRate$ | |
|---|---|---|
| | = 1.0 for $frequency \geq 0.5 \times (1 + Alpha) \times SymbolRate$ | |
| | = [0.5 × (1 + sin[п × ( $frequency$ − $SymbolRate$ /2)/ | |
| $G_{comp}$ | = 1.0 | if $DutyCycle$ =0 |
| | = [0.01 × $DutyCycle$ × sinc(x)]$^{Exponent}$ | if $SincE$ = YES |
| | = [0.01 × $DutyCycle$ × sinc(x)] | if $SincE$ = NO |
| sinc(x) | = sin(x)/x | |
| x | = 0.01 × $DutyCycle$ × п × $frequency$ / $SymbolRate$ | |

5. While Exponent can be any value, the standard value is 1.0 for the ideal raised-cosine filter response or 0.5 to simulate the root raised-cosine filter response when present at both the receiving and transmitting channels.

6. In steady-state frequency-domain analyses, the ideal frequency-domain response described previously is used; however, this ideal response has an infinite duration impulse response that must be approximated for time-domain simulations in either Transient or Circuit Envelope. If DelaySymbols is set too small, then the impulse response will be severely truncated and will not accurately reflect the ideal frequency response.
   Using a Hanning window (WindowType=1) a DelaySymbols value of 15 should result in equivalent frequency domain sidelobes of −90dBc or smaller.

7. Accuracy of this model in Transient or Circuit Envelope can be further controlled through the ImpMaxFreq, ImpDeltaFreq, and ImpMaxPts parameters.

8. The filter can include gain equalization to compensate for duty cycle roll-off. If DutyCycle = 0.0, then no compensation will be applied. If SincE=YES, Exponent will be applied to the gain compensation term, $G_{comp}$. The Exponent term is always present in the $G_{filt}$ term.

9. This component has no default artwork associated with it.

# LPF_Bessel (Lowpass Filter, Bessel-Thompson)

## Symbol



## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Fpass | Passband edge frequency | GHz | 1 |
| Apass | Attenuation at passband edges | dB | 3 |
| GDpass | Group delay rolloff at passband edge, 0 < GDpass < 1 | None | 0.9 |
| StopType | Stopband input impedance type: OPEN or SHORT | None | open |
| MaxRej | Maximum rejection level | dB | None |
| N | Filter order (if N > 0, it overwrites BWstop and Astop)); if not given, it is calculated based on BWpass, Apass and GDpass | None | 0 |
| IL | Passband insertion loss | dB | 0 |
| Qu | Unloaded quality factor for resonators, default setting is an infinite Qu and expresses a dissipationless resonant circuit. | None | 1e308 |
| Z1 | Input port reference impedance | Ohm | 50 |
| Z2 | Output port reference impedance | Ohm | 50 |
| Temp | Temperature | °C | None |

## Range of Usage

Fpass > 0
0.01 ≤ Apass ≤ 3.0
0 < GDpass < 1
1 ≤ N ≤ 15
Qu ≥ 1

## Notes/Equations

1. Refer to *Filter Categories* (ccsys).
2. For information on LPF behavior at DC, refer to *Lowpass Filter Behavior at DC* (ccsys).
3. This component has no default artwork associated with it.

# LPF_Butterworth (Lowpass Filter, Butterworth)

## Symbol



## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Fpass | Passband edge frequency | GHz | 1 |
| Apass | Attenuation at passband edges | dB | 3 |
| Fstop | Stopband edge frequency | GHz | 1.2 |
| Astop | Attenuation at stopband edges | dB | 20 |
| StopType | Stopband input impedance type: OPEN or SHORT | None | open |
| MaxRej | Maximum rejection level | dB | None |
| N | Filter order (if N > 0, it overwrites BWstop and Astop)); if not given, it is calculated based on BWpass, Apass and GDpass | None | 0 |
| IL | Passband insertion loss | dB | 0 |
| Qu | Unloaded quality factor for resonators, default setting is an infinite Qu and expresses a dissipationless resonant circuit. | None | 1e308 |
| Z1 | Input port reference impedance | Ohm | 50 |
| Z2 | Output port reference impedance | Ohm | 50 |
| Temp | Temperature | °C | None |

## Range of Usage

Fpass > 0
0.01 ≤ Apass ≤ 3.0
1 ≤ N ≤ 15
Qu ≥ 1

## Notes/Equations

1. Refer to *Filter Categories* (ccsys).
2. For information on LPF behavior at DC, refer to *Lowpass Filter Behavior at DC* (ccsys).
3. This component has no default artwork associated with it.

# LPF_Chebyshev (Lowpass Filter, Chebyshev)

## Symbol



## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Fpass | Passband edge frequency | GHz | 1 |
| Apass | Attenuation at passband edges | dB | 1 |
| Ripple | Stopband ripple | dB | 1 |
| Fstop | stopband edge frequency | GHz | 1.2 |
| Astop | Attenuation at stopband edges | dB | 20 |
| StopType | Stopband input impedance type: OPEN or SHORT | None | open |
| MaxRej | Maximum rejection level | dB | None |
| N | Filter order (if N > 0, it overwrites BWstop and Astop)); if not given, it is calculated based on BWpass, Apass and GDpass | None | 0 |
| IL | Passband insertion loss | dB | 0 |
| Qu | Unloaded quality factor for resonators, default setting is an infinite Qu and expresses a dissipationless resonant circuit. | None | 1e308 |
| Z1 | Input port reference impedance | Ohm | 50 |
| Z2 | Output port reference impedance | Ohm | 50 |
| Temp | Temperature | °C | None |

## Range of Usage

Fpass > 0
$0.01 \leq Ripple \leq 3.0$
$1 \leq N \leq 15$
$Qu \geq 1$

## Notes/Equations

1. Refer to *Filter Categories* (ccsys).
2. For information on LPF behavior at DC, refer to *Lowpass Filter Behavior at DC* (ccsys).
3. This component has no default artwork associated with it.

# LPF_Elliptic (Lowpass Filter, Elliptic)

## Symbol



## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Fpass | Passband edge frequency | GHz | 1 |
| Ripple | Stopband ripple | dB | 1 |
| Fstop | Stopband edge frequency | GHz | 1.2 |
| Astop | Attenuation at stopband edges | dB | 20 |
| StopType | Stopband input impedance type: OPEN or SHORT | None | open |
| MaxRej | Maximum rejection level | dB | None |
| N | Filter order (if N > 0, it overwrites BWstop and Astop)); if not given, it is calculated based on BWpass, Apass and GDpass | None | 0 |
| IL | Passband insertion loss | dB | 0 |
| Qu | Unloaded quality factor for resonators, default setting is an infinite Qu and expresses a dissipationless resonant circuit. | None | 1e308 |
| Z1 | Input port reference impedance | Ohm | 50 |
| Z2 | Output port reference impedance | Ohm | 50 |
| Temp | Temperature | °C | None |

## Range of Usage

Fpass > 0
$0.01 \leq Ripple \leq 3.0$
Astop > 0
$1 \leq N \leq 15$

## Notes/Equations

1. Refer to *Filter Categories* (ccsys).
2. For information on LPF behavior at DC, refer to *Lowpass Filter Behavior at DC* (ccsys).
3. This component has no default artwork associated with it.

# LPF_Gaussian (Lowpass Filter, Gaussian)

## Symbol



## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Fpass | Passband edge frequency | GHz | 1 |
| Apass | Attenuation at passband edges | dB | 3 |
| GDpass | Group delay rolloff at passband edge, 0 < GDpass < 1 | None | 0.9 |
| StopType | Stopband input impedance type: OPEN or SHORT | None | open |
| MaxRej | Maximum rejection level | dB | None |
| N | Filter order (if N > 0, it overwrites BWstop and Astop)); if not given, it is calculated based on BWpass, Apass and GDpass | None | 0 |
| IL | Passband insertion loss | dB | 0 |
| Qu | Unloaded quality factor for resonators, default setting is an infinite Qu and expresses a dissipationless resonant circuit. | None | 1e308 |
| Z1 | Input port reference impedance | Ohm | 50 |
| Z2 | Output port reference impedance | Ohm | 50 |
| Temp | Temperature | °C | None |

## Range of Usage

Fpass > 0
0.01 ≤ Apass ≤ 3.0
0 < GDpass < 1
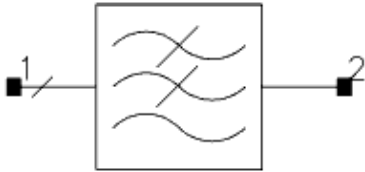1 ≤ N ≤ 15
Qu ≥ 1

## Notes/Equations

1. Refer to *Filter Categories* (ccsys).
2. For information on LPF behavior at DC, refer to *Lowpass Filter Behavior at DC* (ccsys).
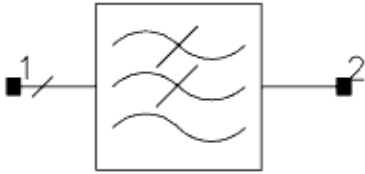3. This component has no default artwork associated with it.

# LPF_GMSK (Lowpass Filter, GMSK)

## Symbol
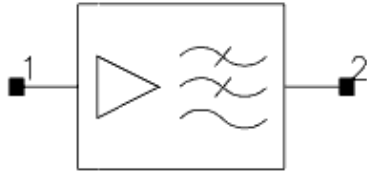


## Parameters

| Name | Description | Units | Default |
|---|---|---|---|
| BT | Bandwidth-Time Product of Gaussian Filter | None | 0.3 |
| BitRate | Digital bit rate defining filter bandwidth | kHz | 270.833 |
| DelayBits | Number of bits delayed by filter | None | 5 |
| Gain | Gain factor | None | 1.0 |
| Zout | Output impedance | Ohm | 50 |
| WindowType | Window type applied to impulse response: 0=None, 1=Hann, 2=Hamming | None | 0 |
| ImpMaxFreq | Maximum frequency to consider when calculating impulse response | | |
| ImpDeltaFreq | Frequency sample spacing when calculating impulse response | | |
| ImpMaxPts | Maximum number of points in impulse response | None | None |

## Range of Usage

$0 \leq$ Alpha $\leq 1$
DelayBits $\geq 1$

## Notes/Equations/Reference

1. Refer to *Filter Categories* (ccsys).
2. For information on lowpass filter behavior at DC, refer to *Lowpass Filter Behavior at DC* (ccsys).
3. This filter is unidirectional. Its input impedance is infinite and its output impedance is specified by Zout.
4. This Gaussian filter is used in the GMSK modulation system.
5. In the steady-state frequency domain analysis, an ideal frequency-domain response is used. However, this ideal response has an infinite duration impulse response that must be approximated for time domain simulations in either Transient or Circuit Envelope. If DelayBits is set too small, then the impulse response will be severely truncated and will not accurately reflect the ideal frequency response.
6. A value of 1.0 will be used internally when DelayBits is set to a value that is less than 1.0.

7. Accuracy of this model in Transient or Circuit Envelope can be further controlled through the ImpMaxFreq, ImpDeltaFreq, and ImpMaxPts parameters.
8. This component has no default artwork associated with it.

# LPF_PoleZero (Lowpass Filter, Pole Zero)

## Symbol
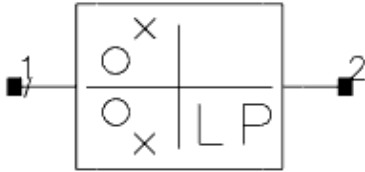


## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Poles | List of complex poles | None | list(−0.7 + j × 0.7, |
| Zeros | List of complex zeros | None | None |
| Gain | Gain factor | None | 1.0 |
| Fpass | Passband edge frequency | GHz | 1 |
| StopType | Stopband input impedance type: OPEN or SHORT | None | open |
| Z1 | Input port reference impedance | Ohm | 50 |
| Z2 | Output port reference impedance | Ohm | 50 |

## Notes/Equations

1. This is an S-domain filter.
2. Poles and Zeros are a list of complex pole/zero locations.
   The transfer function for the filter is:

$$S_{21} = Gain\left(\frac{(S - Zero1)(S - Zero2)...}{(S - Pole1)(S - Pole2)...}\right)$$

   where
   S = j(Freq / Fpass)
   and
   Freq is the analysis frequency
   At least one pole must be supplied.
3. For information on lowpass filter behavior at DC, refer to *Lowpass Filter Behavior at DC* (ccsys).

# LPF_Polynomial (Lowpass Filter, Polynomial)

**Symbol**



**Parameters**

| Name | Description | Units | Default |
|---|---|---|---|
| Numerator | List of numerator coefficients | None | list(1) |
| Denominator | List of denominator coefficients | None | list(1,1.4,1) |
| Gain | Gain factor | None | 1.0 |
| Fpass | Passband edge frequency | GHz | 1 |
| StopType | Stopband input impedance type: OPEN or SHORT | None | open |
| Z1 | Input port reference impedance | Ohm | 50 |
| Z2 | Output port reference impedance | Ohm | 50 |

**Notes/Equations**

1. This is an S-domain filter.
2. Denominator and Numerator are a list of polynomial coefficients.
   The transfer function for the filter is:

   $$S_{21} = Gain\left(\frac{N_0 + (N_1 \times S) + (N_2 \times S^2)\dots}{D_0 + (D_1 \times S) + (D_2 \times S^2)\dots}\right)$$

   where
   S = j(Freq / Fpass)
   and
   Freq is the analysis frequency
   At least one denominator coefficient must be supplied.
3. For information on lowpass filter behavior at DC, refer to *Lowpass Filter Behavior at DC* (ccsys).

179

# LPF_RaisedCos (Lowpass Filter, Raised-Cosine)

## Symbol



## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Alpha | Rolloff factor defining filter excess bandwidth, 0 ≤ Alpha ≤ 1 | None | 0.35 |
| SymbolRate | Digital symbol rate defining filter bandwidth | kHz | 24.3 |
| DelaySymbols | Number of symbols delayed by filter | None | 5 |
| Exponent | Exponent factor ( 0 ≤ Exponent ≤ 1 ), to provide for Root Raised-Cosine filter | None | 0.5 |
| DutyCycle | Pulse duty cycle in percent, used for sinc(x) correction | None | 0 |
| SincE | Flag to include the Exponent factor on the sinc(x) correction: yes or no | None | no |
| Gain | Gain factor | None | 1.0 |
| Zout | Output impedance | Ohm | 50 |
| WindowType | Window type applied to impulse response: 0=None, 1=Hann, 2=Hamming | None | 0 |
| ImpMaxFreq | Maximum frequency to consider when calculating impulse response | | |
| ImpDeltaFreq | Frequency sample spacing when calculating impulse response | | |
| ImpMaxPts | Maximum number of points in impulse response | None | None |
| Other | output string to netlist | None | None |

## Range of Usage

0 ≤ Alpha ≤ 1
DelaySymbols ≥ 1
0 ≤ Exponent ≤ 1
0 ≤ DutyCycle ≤ 100

## Notes/Equations

1. Refer to *Filter Categories* (ccsys).
2. For information on lowpass filter behavior at DC, refer to *Lowpass Filter Behavior at DC* (ccsys).
3. This filter is unidirectional; input impedance is infinite; output impedance is specified by Zout.
4. Voltage gain is described by the following function.

$$\frac{V_{out}}{V_{in}} = \frac{Gain \times G_{filt}}{G_{comp}} e^{-j2\pi frequency\left(\frac{DelaySymbols}{SymbolRate}\right)}$$

where:

| $G_{filt}$ | = 1.0 for *frequency* ≤ 0.5 × (1 − *Alpha* ) × *SymbolRate* | |
|---|---|---|
| | = 0.0 for *frequency* ≥ 0.5 × (1 + *Alpha* ) × *SymbolRate* | |
| | = [0.5 × (1 − sin[п × ( *frequency* − *SymbolRate* /2)/ | |
| $G_{comp}$ | = 1.0 | if *DutyCycle* =0 |
| | = [0.01 × *DutyCycle* × sinc(x)]$^{Exponent}$ | if *SincE* = YES |
| | = [0.01 × *DutyCycle* × sinc(x)] | if *SincE* = NO |
| sinc(x) | = sin(x)/x | |
| x | = 0.01 × *DutyCycle* × п × *frequency* / *SymbolRate* | |

5. While Exponent can be any value, the standard value is 1.0 for the ideal raised-cosine filter response or 0.5 to simulate the root raised-cosine filter response when present at both the receiving and transmitting channels.

6. In steady-state frequency-domain analyses, the ideal frequency-domain response described previously is used; however, this ideal response has an infinite duration impulse response that must be approximated for time-domain simulations in either transient or circuit envelope. If DelaySymbols is set too small, then the impulse response will be severely truncated and will not accurately reflect the ideal frequency response.
   A DelaySymbols value of 15 should result in saturated frequency-domain sidelobes of -75dBc or smaller. This number is approximate and represents the saturated sidelobe level at frequencies far greater than the filter's cutoff frequency. The sidelobes at, say, twice the filter's cutoff frequency have generally not saturated and will typically be higher than -75 dBc. The saturated sidelobe level may depend on whether a transient or circuit envelope simulation is performed and on the window type used. It is significantly lower than -75dBc in many cases. The accuracy of this model in transient or circuit envelope simulations can be further controlled through the ImpMaxFreq, ImpDeltaFreq, and ImpMaxPts parameters.

7. The filter can include gain equalization to compensate for duty cycle roll-off. If DutyCycle = 0.0, then no compensation will be applied. If SincE=YES, Exponent will be applied to the gain compensation term $G_{comp}$ . The Exponent term is always present in the $G_{filt}$ term.

8. This component has no default artwork associated with it.

# SAW_Filter (Saw Filter)

## Symbol
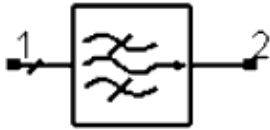


## Parameters

| Name | Description | Units | Default |
|---|---|---|---|
| Fcenter | Stopband center frequency | MHz | 300 |
| IL | Passband insertion loss | dB | 15 |
| BWpass | Passband edge-to-edge width | MHz | 50 |
| Apass | Attenuation at passband edges | dB | 1 |
| BWstop | Stopband edge-to-edge width | MHz | 70 |
| Astop | Attenuation at stopband edges | dB | 40 |
| GDelay | Group delay | sec | 0.4 |
| Aripple | Passband amplitude ripple | dB | 0.5 |
| PhaRipple | Passband phase ripple from linear phase | deg | None |
| MaxRej | Maximum rejection level | dB | |
| Zin | Input impedance | Ohm | 50 |
| Zout | Output impedance | Ohm | 50 |
| WindowType | Window type applied to impulse response: 0=None, 1=Hann, 2=Hamming | None | 0 |
| Temp | Temperature | °C | None |
| Other | Output string to netlist | None | None |

## Range of Usage

BWstop ≥ BWpass

GDelay ≥ π/BWstop

## Notes/Equations

1. SAW_Filter models the typical behavior of SAW bandpass filters. If the PhaRipple parameter is not specified, the filter will have perfect linear phase.
2. To maintain causality, Gdelay must be set to at least π/BWstop.

# Modulators and Demodulators

## Introduction

The *Filters - < filter type >* and *System - < device type >* palettes contain two fundamentally different types of behavioral system models.

*Filters* , *System - Amps & Mixers* , and *System - Mod/Demod* can be classified as *tops-down system models* that support a tops-down system design flow where model behaviors are characterized by a small number of independent parameters such as frequency, power and load. They are often referred to as *parameter-based behavioral models* .

*System - Data Models* can be classified as *bottoms-up system models* that support a bottoms-up verification flow where model behaviors are extracted from a simulation (or measurement) of a transistor-level circuit. They are often referred to as *data-based behavioral models* .

The *parameter-based behavioral models* typically provide superior speed relative to the data-based behavioral models with both of these being vastly superior to a brute-force transistor-level simulation.

The *data-based behavioral models* typically provide superior accuracy relative to the *parameter-based behavioral models* as they capture actual behaviors of implemented circuit components and not just design specifications.

The differences between *parameter* - and *data-based behavioral models* justify a palette emphasis on flow (all data-based behavioral models grouped together) rather than functionality (all amplifiers, mixers, modulators, and demodulators grouped together) and resulted in the addition of a *System - Data Models* palette.

The use model for parameter-based behavioral models is to simply set a series of parameters prior to using the model. The use model for data-based behavioral models is slightly more involved. For a discussion, refer to Chapter 8, System Data Models.

The modulators and demodulators system model library contains time domain tuned modulators and tuned demodulators. Each component in this library is described following this introduction.

When using the System-Mod/Demod components in the Analog/RF schematic, it is important to note that terminating the component with a load resistance equivalent to the component output resistance will provide an output voltage that is half of the applied input voltage. Consider the Thevenin equivalent of the output of a Mod/Demod component.

Mod/Demod Component
Output Resistance
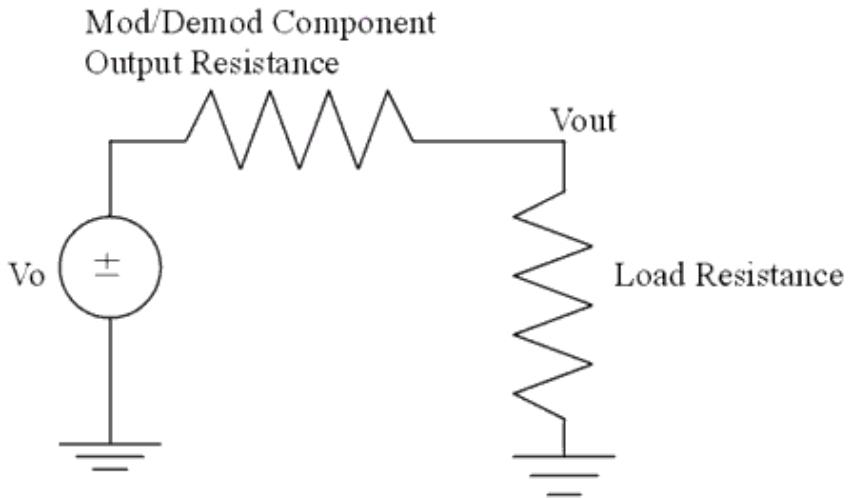


In an Analog/RF schematic, the value for Vout will be 1/2 Vo when Output Resistance=Load Resistance. In general, Vout=Vo × Load Resistance/(Load Resistance + Output Resistance). Thus, this is the potential divider action. All of the components in the System-Mod/Demod library have this property except for N_StateDemod and PM_UnwrapDemodTuned. (These two components do not have an Rout parameter).

For a similar circuit in a DSP schematic, the effect of the potential divider on the output voltage will not be noticeable when the output resistance equals the load resistance. In this case, Vout equals Vo. For the DSP components, there is an additional factor of 2 at the output voltage to cancel the factor of 1/2 from the potential divider. For a description of the digital implementation of the modulators and demodulators, and the potential divider action, refer to *Introduction: Timed, Modem Components* in the Signal Processing component documentation.

## Components

- *AM DemodTuned (AM Demodulator, Tuned)* (ccsys)
- *AM ModTuned (AM Modulator, Tuned)* (ccsys)
- *FM DemodTuned (FM Demodulator, Tuned)* (ccsys)
- *FM ModTuned (FM Modulator, Tuned)* (ccsys)
- *IQ DemodTuned (I-Q Demodulator, Tuned)* (ccsys)
- *IQ ModTuned (I-Q Modulator, Tuned)* (ccsys)
- *N StateDemod (N-State Demodulator)* (ccsys)
- *N StateMod (N-State Modulator)* (ccsys)
- *PI4DQPSK ModTuned (PI-4 DQPSK Modulator, Tuned)* (ccsys)
- *PM DemodTuned (PM Demodulator, Tuned)* (ccsys)
- *PM ModTuned (PM Modulator, Tuned)* (ccsys)
- *PM UnwrapDemodTuned (PM Unwrapped Demodulator, Tuned)* (ccsys)
- *QPSK ModTuned (QPSK Modulator, Tuned)* (ccsys)

# AM_DemodTuned (AM Demodulator, Tuned)
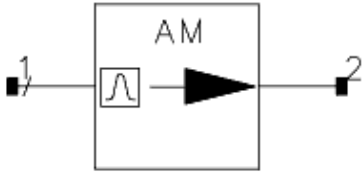
## Symbol



## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Fnom | Nominal input frequency | GHz | 1 |
| Rout | Output resistance | Ohm | 50 |

## Notes/Equations

1. This is a tuned demodulator that selects the input harmonic closest to the specified Fnom frequency and generates a baseband output signal equal to the instantaneous amplitude of the selected carrier frequency.
   If there is no analysis harmonic frequency close enough to the Fnom frequency, a warning is issued and the output is 0. Input impedance is infinite; output impedance is set by Rout, and is limited to a minimum value of 0.1 Ohm. The open circuit output voltage is defined by

   $$Vout_0 = mag(Vin_k)$$

   where

   $$mag(V) = \sqrt{Re(V)^2 + Im(V)^2}$$

   and
   $k$ represents the value at the analysis harmonic frequency closest to the Fnom value
   0 represents the baseband component of the output voltage. All non-baseband output frequency components are 0.

2. This model generates only the absolute value of the input because the imaginary part of baseband signals is 0 for baseband envelope simulations. It does not produce the required baseband signal in transient simulation so you have to add a filter after it to filter the unwanted signal.

# AM_ModTuned (AM Modulator, Tuned)

## Symbol
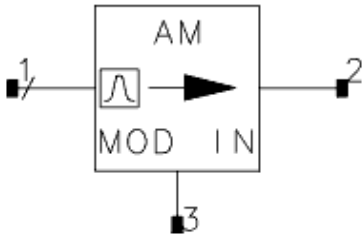


## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| ModIndex | Modulation index | None | 1.0 |
| Fnom | Nominal input frequency | GHz | 1 |
| Rout | Output resistance | Ohm | 50 |

## Notes/Equations

1. This is a tuned modulator that selects the input harmonic defined by the specified Fnom frequency and amplitude modulates it by the baseband modulation input.
   If there is no analysis harmonic frequency close enough to the Fnom frequency, a warning is issued and the output is 0. The two input impedances are infinite. The output impedance is set by Rout, and is limited to a minimum value of 0.1 Ohm. The open circuit output voltage is defined by
   $$Vout_k = (1 + ModIndex \times Vmod_0) \times Vin_k$$
   where
   $k$ represents the value at the analysis harmonic frequency closest to the Fnom value
   0 represents the baseband component of the modulation input voltage.
2. The RF carrier is injected at pin 1; the modulating signal is injected at pin 3. The resulting AM signal is present at pin 2. The RF carrier should be a frequency-domain source; the modulating signal should be a time-domain source.
3. This model works in transient and baseband envelope simulations. Because the describing equation is valid for baseband signals, Fnom has no effect.

# FM_DemodTuned (FM Demodulator, Tuned)

## Symbol



## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Sensitivity | Demodulation sensitivity, in Hz/Volt | kHz | 1 |
| Fnom | Nominal input frequency | GHz | 1 |
| Rout | Output resistance | Ohm | 50 |

## Notes/Equations

1. This is a tuned demodulator that selects the input harmonic closest to the specified Fnom frequency and generates a baseband output signal equal to the instantaneous frequency of the selected carrier frequency.

   If there is no analysis harmonic frequency close enough to the Fnom frequency, a warning is issued and the output is 0. Input impedance is infinite; output impedance is set by Rout, and is limited to a minimum value of 0.1 Ohm.

   The instantaneous frequency difference is equal to the time derivative of the instantaneous phase. This is approximated using a simple finite difference. As long as the phase does not change by greater than 180° in one time step, the frequency calculation is non-ambiguous. The open circuit output voltage can be described as:

   $$Vout_0 = \frac{\Phi(Vin_k(t)) - \Phi(Vin_k(t-timestep))}{2\pi \times Sensitivity \times timestep}$$

   where

   $$\Phi(V) = atan\frac{Im(V)}{Re(V)}$$

   $k$ represents the value at the analysis harmonic frequency closest to the Fnom value
   0 represents the baseband component of the output voltage.
   All non-baseband output frequency components are 0.

2. For transient and baseband envelope simulations, this model does not generate the expected results because the imaginary part of baseband signals is 0.

# FM_ModTuned (FM Modulator, Tuned)
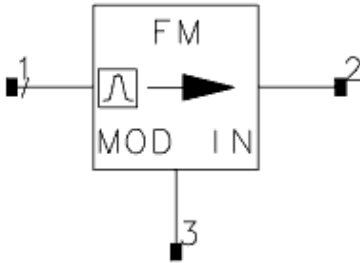
## Symbol



## Parameters

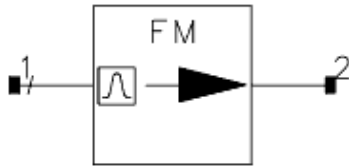| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Sensitivity | Modulation sensitivity, in Hz/Volt | kHz | 1 |
| Fnom | Nominal input frequency | GHz | 1 |
| Rout | Output resistance | Ohm | 50 |

## Notes/Equations

1. This is a tuned modulator that selects the input harmonic defined by the specified Fnom frequency and frequency modulates it by the baseband modulation input.
   If there is no analysis harmonic frequency close enough to the Fnom frequency, a warning is issued and the output is 0. The two input impedances are infinite. The output impedance is set by Rout, and is limited to a minimum value of 0.1 Ohm. The open circuit output voltage is defined by

   $$Vout_k = e^{j2\pi Sensitivity \int_0^t Vm_0 dt} \times Vin_k$$

   where
   $k$ represents the value at the analysis harmonic frequency closest to the Fnom value
   0 represents the baseband component of the modulation input voltage.
   For time t=0, the integrator of the modulation input signal is reset to 0. In all steady-state analyses such as harmonic balance, dc, and ac, time is kept at 0, so no modulation occurs.
2. The RF carrier is injected at pin 1; the modulating signal is injected at pin 3. The resulting FM signal is present at pin 2. The RF carrier should be a frequency-domain source; the modulating signal should be a time-domain source.
3. This model will not function as a frequency modulator in transient and baseband envelope simulations because of the use of the complex exponential operator.

# IQ_DemodTuned (I/Q Demodulator, Tuned)
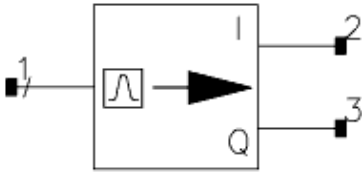


## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Fnom | Nominal input frequency | GHz | 1 |
| Rout | Output resistance | Ohm | 50 |

## Notes/Equations

1. This is a tuned demodulator that selects the input harmonic closest to the specified Fnom frequency and generates two baseband output signals equal to the instantaneous in-phase and quadrature-phase components of the selected carrier frequency.
   If there is no analysis harmonic frequency close enough to the Fnom frequency, a warning is issued and the output is 0. Input impedance is infinite; output impedance is set by Rout, and is limited to a minimum value of 0.1 Ohm. The open circuit output voltages are defined simply by

   $$Vi_0 = Re(Vin_k)$$
   $$Vq_0 = Im(Vin_k)$$

   where
   $k$ represents the value at the analysis harmonic frequency closest to the Fnom value
   0 represents the baseband component of the output voltages.
   All non-baseband output frequency components are 0.
2. In transient and baseband envelope simulations, because the imaginary part of baseband signals is 0, this model replicates the input on the I output only.

# IQ_ModTuned (I/Q Modulator, Tuned)
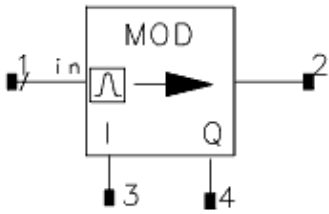


## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Fnom | Nominal input frequency | GHz | 1 |
| Rout | Output resistance | Ohm | 50 |

## Notes/Equations

1. This is a tuned modulator that selects the input harmonic defined by the specified Fnom frequency and modulates it according to the I (in-phase) and Q (quadrature) modulation inputs.
   If there is no analysis harmonic frequency close enough to the Fnom frequency, a warning is issued and the output is 0. All three input impedances are infinite; output impedance is set by Rout, and is limited to a minimum value of 0.1 Ohm. The open circuit output voltage is defined by

   $$Vout_k = (V_{I0} + j \times V_{Q0}) \times Vin_k$$

   where
   *k* represents the value at the analysis harmonic frequency closest to the Fnom value
   0 represents the baseband component of the two modulation input voltages.
2. The RF carrier is injected at pin 1; I data is injected at pin 3; Q data is injected at pin 4. The resulting modulated signal is present at pin 2. The RF carrier should be a frequency-domain source; the modulating signals should be a time-domain source.
3. This model will not function as a phase modulator in transient and baseband envelope simulation because of the use of the complex j operator.
4. Both AM and PM modulation can be generated.

# N_StateDemod (N-State Demodulator)

## Symbol
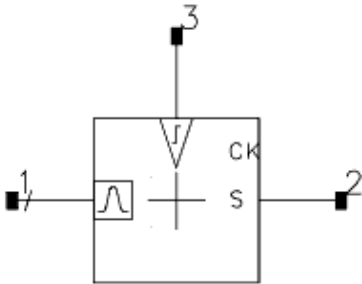


## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Fnom | Nominal input frequency | GHz | 1 |
| StateArray | Complex array of state values | None | list(1+j,-1+j,-1-j,1-j) |

## Notes/Equations

1. This is a tuned demodulator that selects the input harmonic closest to the specified Fnom frequency and generates a baseband output state signal representing the nearest complex nominal state of the selected carrier at the time of last rising clock edge.
   If there is no analysis harmonic frequency close enough to the Fnom frequency, a warning is issued and the output is 0. On each rising clock edge, the I/Q information of the carrier is sampled. Using a Euclidean distance measure, the closest nominal state in the StateArray variable is determined and that state number, minus 1, is output as the baseband state voltage. This can be considered a 2-dimensional quantizer.
2. The nominal state locations are arbitrary and user definable in the StateArray variable, which can be created from a list of complex values using the list( ) function.
3. In transient and baseband envelope simulations, this state demodulation model functions correctly and could be used as a 1-dimensional arbitrary state quantizer.

# N_StateMod (N-State Modulator)

## Symbol
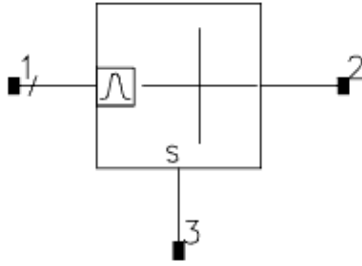


## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| MaxStates | Maximum number of input states | None | 4 |
| StateArray | Complex array of state values | None | list(1+j,-1+j,-1-j,1-j) |
| Fnom | Nominal input frequency | GHz | 1 |
| Rout | Output resistance | Ohm | 50 |

## Notes/Equations

1. This N-state modulator can be used to create an arbitrary, user-defined constellation of complex modulation states. The input harmonic closest to the specified Fnom frequency is selected and modulated by a sequence of complex states.
   If there is no analysis harmonic frequency close enough to the Fnom frequency, a warning is issued and the output is 0. The states are defined by the complex array variable StateArray. The sequence of states is determined by the baseband component of the state modulation input. The two input impedances are infinite; output impedance is set by Rout, and is limited to a minimum value of 0.1 Ohm. The open circuit output voltage is defined by

$$Vout_k = State[int(Vs_0) + 1] \times Vin_k$$

   where
   $k$ represents the value at the analysis harmonic frequency closest to the Fnom value
   0 represents the baseband component of the modulation state input voltage; this input voltage is internally limited to be between 0 and MaxStates $-1$.
   The StateArray variable can be created from list of complex values using the list( ) function.
2. This model will not function as a complex modulator in transient and baseband envelope simulations if complex state values are specified.
3. 16-State Modulation Example and Simulation Results show a schematic example and simulation results for a 16-state modulator being swept linearly through all 16 of its states.

VAR
VAR1
myqam16=list(1,3,2-j*1,1-j*2,-3*j,-1-j*2,-2-j*1,-3,-2+j*1,-1+j*2,3*j,1+2*j,2+1*j,1*j,-1,-1*j)
f1=100MHz
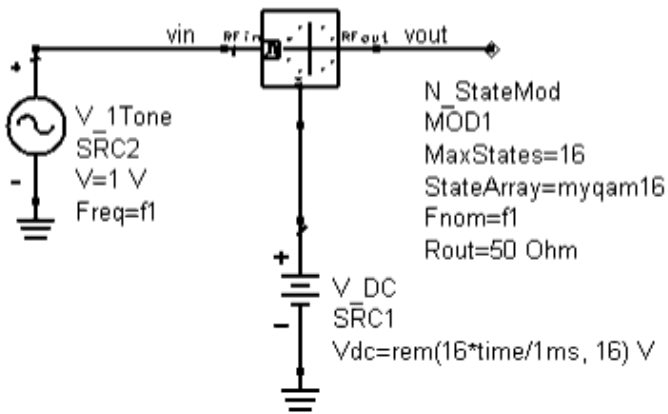
ENVELOPE

Envelope
Env1
Freq[1]=f1
Order[1]=1
Stop=1msec
Step=10 usec

vin    RFin        RFout  vout

V_1Tone
SRC2
V=1 V
Freq=f1

N_StateMod
MOD1
MaxStates=16
StateArray=myqam16
Fnom=f1
Rout=50 Ohm

V_DC
SRC1
Vdc=rem(16*time/1ms, 16) V

## 16-State Modulation Example

Eqn Iout=real(vout[1])      Eqn Qout=imag(vout[1])
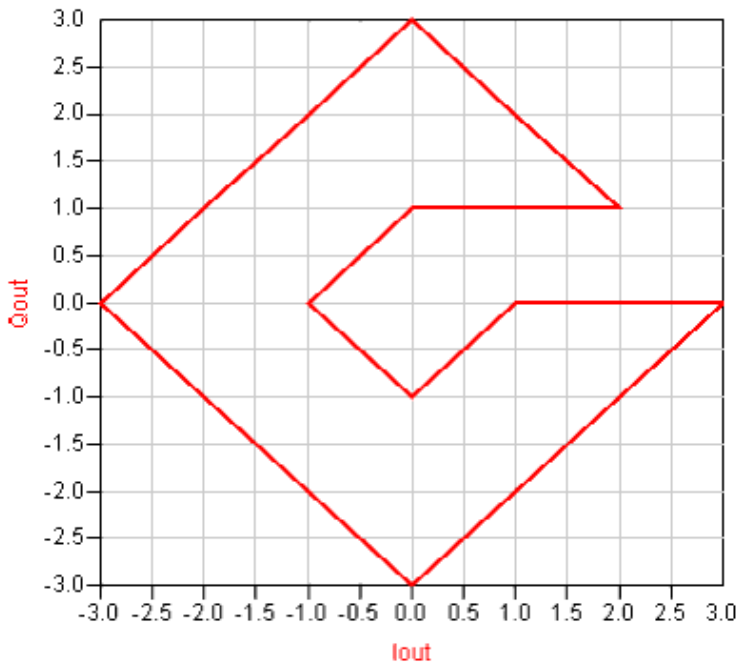


## Simulation Results

# PI4DQPSK_ModTuned (PI-4 DQPSK Modulator, Tuned)
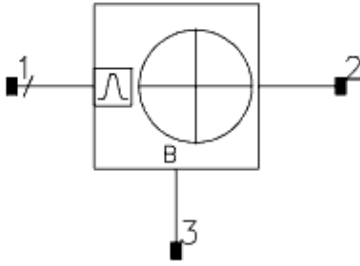


## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Fnom | Nominal input frequency | GHz | 1 |
| Rout | Output resistance | Ohm | 50 |
| SymbolRate | Output symbol rate (1/2 input bit rate) | kHz | 24.3 |
| Delay | Sampling delay | nsec | 50 |

## Notes/Equations

1. This tuned PI/4 DQPSK modulator selects the input harmonic closest to the specified Fnom frequency and modulates it according to the phase state determined by differentially encoding the input bit stream and applying the PI/4 phase offset.
   If there is no analysis harmonic frequency close enough to the Fnom frequency, a warning is issued and the output is 0. The input bit stream is sampled at a rate determined by the SymbolRate with an initial synchronizing delay determined by the Delay parameter. The actual sampling rate is rounded to an integer multiple of the system timestep-use a corresponding discrete time source to generate this serial bit stream or ensure that the analysis sampling rate is an integer multiple of the bit rate. Two consecutive bits are monitored, along with the previous phase state, to determine the next phase state. An input bit is assumed to be 1 if it is greater than 0.5V and 0 otherwise.
2. The RF carrier is injected at pin 1; the input bit stream is injected at pin 3. The resulting PI4DQPSK signal is present at pin 2. The RF carrier should be a frequency-domain source; the modulating signal should be a time-domain source.
3. The input impedances of both the serial bit stream input and the RF carrier are infinite; output impedance is set by Rout, and is limited to a minimum value of 0.1 Ohm. The open circuit output voltage is defined by:

$$Vout_k = e^{jNextPh(Vm_0(t),\ Vm_0(t-BitTime),\ CurrentPh)} \times Vin_k$$

   where
   k represents the value at the analysis harmonic frequency closest to the Fnom value

| $Vm_0(t)$ | $Vm_0(t - BitTime)$ | Phase Transition = Next Phase − Current Phase |
|---|---|---|
| 0 | 0 | 45° |
| 0 | 1 | 135° |
| 1 | 0 | −45° |
| 1 | 1 | −135° |

4. This model will not function as a modulator in transient and baseband envelope simulations because of the use of complex exponential functions.

# PM_DemodTuned (PM Demodulator, Tuned)
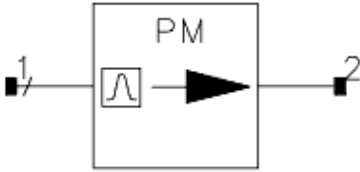
## Symbol



## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Sensitivity | Demodulation sensitivity, in deg/Volt | deg | 180/pi |
| Fnom | Nominal input frequency | GHz | 1 |
| Rout | Output resistance | Ohm | 50 |

## Notes/Equations

1. This is a tuned demodulator that selects the input harmonic closest to the specified Fnom frequency and generates a baseband output signal equal to the instantaneous phase of the selected carrier frequency.
   If there is no analysis harmonic frequency close enough to the Fnom frequency, a warning is issued and the output is 0. The input impedance is infinite. The output impedance is set by Rout, and is limited to a minimum value of 0.1 Ohm. The open circuit output voltage is defined by

   $$Vout_0 = \frac{\Phi(Vin_k(t)) \times 180}{\pi \times Sensitivity}$$

   where

   $$\Phi(V) = atan\frac{Im(V)}{Re(V)}$$

   $k$ represents the value at the analysis harmonic frequency closest to the Fnom value
   0 represents the baseband component of the output voltage.
   All non-baseband output frequency components are 0.
   The maximum phase range of this demodulator is ±180°.

2. In transient and baseband envelope simulations, this model does not generate the expected results because the imaginary part of baseband signals is 0.

# PM_ModTuned (PM Modulator, Tuned)
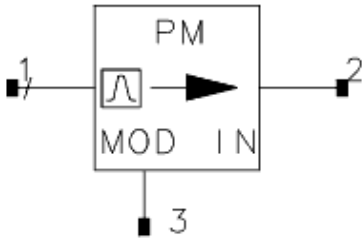
## Symbol



## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Sensitivity | Modulation sensitivity, in deg/Volt | deg | 10 |
| Fnom | Nominal input frequency | GHz | 1 |
| Rout | Output resistance | Ohm | 50 |

## Notes/Equations

1. This is a tuned modulator that selects the input harmonic defined by the specified Fnom frequency and phase modulates it by the baseband modulation input. If there is no analysis harmonic frequency close enough to the Fnom frequency, a warning is issued and the output is 0. The two input impedances are infinite. The output impedance is set by Rout, and is limited to a minimum value of 0.1 Ohm. The open circuit output voltage is defined by

$$Vout_k = e^{j\pi Sensitivity \times (Vmod_0)/180} \times Vin_k$$

   where
   $k$ represents the value at the analysis harmonic frequency closest to the Fnom value
   0 represents the baseband component of the modulation input voltage.
2. The RF carrier is injected at pin 1; the modulating signal is injected at pin 3. The resulting PM signal is present at pin 2. The RF carrier should be a frequency-domain source; the modulating signal should be a time-domain source.
3. This model will not function as a phase modulator in transient and baseband envelope simulations because of the use of the complex exponential operator.

# PM_UnwrapDemodTuned (PM Unwrapped Demodulator, Tuned)
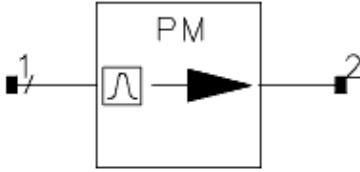
## Symbol



## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Sensitivity | Demodulation sensitivity, in deg/Volt | deg | 10 |
| Fnom | Nominal input frequency | GHz | 1 |
| MaxAngle | Unwrapped phase angle range (+/− MaxAngle) | deg | 3600 |

## Notes/Equations

1. This is a tuned demodulator that selects the input harmonic closest to the specified Fnom frequency and generates a baseband output signal equal to the instantaneous unwrapped phase of the selected carrier frequency.
   If there is no analysis harmonic frequency close enough to the Fnom frequency, a warning is issued and the output is 0. The input impedance is infinite. The output impedance is 0.
2. The phase range of this demodulator is equal to ± MaxAngle. The phase at time 0 is set equal to the normal 180° phase of the selected carrier frequency. By tracking modulo 360° phase transitions, the unwrapped phase, relative to the initial time 0 value, is calculated. The user can then set the desired modulo range with the MaxAngle parameter. A large number will generate totally unwrapped phase.
   This unwrapping of phase works for time sweeps in envelope analyses; it cannot be used to unwrap the phase of frequency or other parameter sweeps.
3. In transient and baseband envelope simulations, this model does not generate the expected results because the imaginary part of baseband signals is 0.

# QPSK_ModTuned (QPSK Modulator, Tuned)

## Symbol
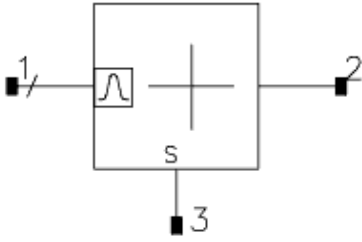


## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Fnom | Nominal input frequency | GHz | 1 |
| Rout | Output resistance | Ohm | 50 |
| State1 | Complex modulation coordinates of State 1 | None | exp(−j × 1 × pi/4) |
| State2 | Complex modulation coordinates of State 2 | None | exp(−j × 3 × pi/4) |
| State3 | Complex modulation coordinates of State 3 | None | exp(−j × 5 × pi/4) |
| State4 | Complex modulation coordinates of State 4 | None | exp(−j × 7 × pi/4) |

## Notes/Equations

1. This tuned quadrature phase shift keying (QPSK) modulator is actually an arbitrary 4-state modulator. The input harmonic closest to the specified Fnom frequency is selected and modulated by one of four user-defined complex values. If there is no analysis harmonic frequency close enough to the Fnom frequency, a warning is issued and the output is 0.
   The complex values are determined by the baseband component of the state modulation input. An input less than 0.5V selects State 1; less than 1.5V selects State 2; less than 2.5V selects State 3; and, any input greater than 2.5V selects State 4. The two input impedances are infinite. The output impedance is set by Rout and is limited to a minimum value of 0.1 Ohm. The open circuit output voltage is defined by:

$$Vout_k = State[int(Vs_0 + 1.5)] \times Vin_k$$

   where
   k represents the value at the analysis harmonic frequency closest to the Fnom value
   0 represents the baseband component of the modulation state input voltage.
2. The RF carrier is injected at pin 1; the input bit stream is injected at pin 3. The resulting QPSK signal is present at pin 2. The RF carrier should be a frequency-domain source; the modulating signal should be a time-domain source.
3. This model will not function as a complex modulator in transient and baseband envelope simulations if the specified state values are complex.

# Passive System Components

- *AntLoad (Antenna Load)* (ccsys)
- *Attenuator (Attenuator)* (ccsys)
- *Balun3Port (Balun, 3-port)* (ccsys)
- *Balun4Port (Balun, 4-port)* (ccsys)
- *Balun6Port (Balun, 6-port)* (ccsys)
- *Circulator (Ideal 3-Port Circulator)* (ccsys)
- *CouplerDual (Dual Coupler)* (ccsys)
- *CouplerSingle (Single Coupler)* (ccsys)
- *Gyrator (Gyrator)* (ccsys)
- *Hybrid90 (Ideal 90-degree Hybrid Coupler)* (ccsys)
- *Hybrid180 (Ideal 180-degree Hybrid Coupler)* (ccsys)
- *IsolatorSML (SMLIsolator)* (ccsys)
- *LOS Link (Line-Of-Sight Antenna Link)* (ccsys)
- *Pad (Pi or Tee Format)* (ccsys)
- *PhaseShiftSML (Phase Shifter)* (ccsys)
- *PwrSplit2 (2-Way Power Splitter)* (ccsys)
- *PwrSplit3 (3-Way Power Splitter)* (ccsys)
- *TimeDelay (Time Delay)* (ccsys)
- *Transformer (Ideal 4-Port Transformer)* (ccsys)
- *TransformerG (Transformer with Ground Reference)* (ccsys)
- *TwoPort (2-Port Model)* (ccsys)

# AntLoad (Antenna Load)

## Symbol



## Illustration



Dipole Mode                    Monopole Mode
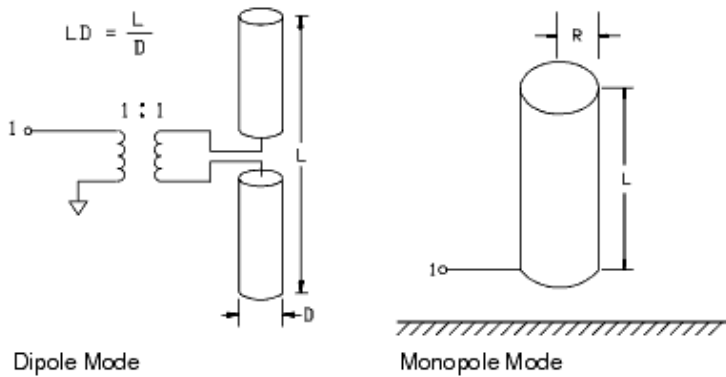
## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| AntType | Antenna type: MONOPOLE or DIPOLE | None | MONOPOLE |
| Length | Physical antenna length | mil | 4e+004 |
| RatioLR | Length-to-radius ratio | None | 10 |

## Range of Usage

$$\text{monopole simulation frequency (MHZ)} \leq \frac{287}{L(meters)} \times \frac{RatioLR}{RatioLR + 1}$$

$$\text{dipole simulation frequency (MHZ)} \leq \frac{575}{L(meters)} \times \frac{RatioLR}{RatioLR + 2}$$

## Notes/Equations

1. This component models the input impedance of a monopole or dipole antenna. Transmission is not modeled.
2. For time-domain analysis, the frequency-domain analytical model is used.
3. This component has no default artwork associated with it.

**References**

1. Constantine A. Balanis, *Antenna Theory: Analysis and Design,* Harper and Row, 1982, pp. 112, 295, 334-335.
2. R. I. Hansen, "Fundamental Limitations in Antennas," *Proceedings of the IEEE* , Vol. 68, No. 2, February, 1981, pp. 170-182.
3. John D. Kraus, *Antennas* , McGraw-Hill Book Company, Inc., 1950, pp. 276-278.

# Attenuator (Attenuator)

## Symbol
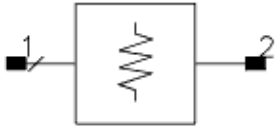


## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Loss | Attenuation | dB | 0 |
| VSWR | Voltage standing wave ratio for both ports | None | 1 |
| Rref | Reference resistance for both ports | Ohm | 50 |
| Temp | Temperature | °C | None |
| CheckPassivity | Check passivity flag: if set to yes, a passivity check is performed and a warning is output if the device is not passive; if set to no, a passivity check is not performed. | None | yes |
| ReturnPhase | Phase characteristic of return loss | None | 0 |

## Range of Usage

VSWR>1.0

## Notes/Equations

1. The S-parameters for Attenuator are:

   $|S12| = |S21| = 10^{(-|Loss|/20)}$
   $Phase(S12) = Phase(S21) = 0$
   $|S11| = |S22| = (VSWR-1) / (VSWR+1)$ (regarding limiting, refer to Note 2)
   $Phase(S11) = Phase(S22) = -90$ if $ReturnPhase < 0$
   $Phase(S11) = Phase(S22) = 0$ if $ReturnPhase = 0$
   $Phase(S11) = Phase(S22) = 90$ if $ReturnPhase > 0$

2. When $ReturnPhase = 0$, $|S11|$ and $|S22|$ will be limited to
   $min(|S11|, 1 - |S21|)$
   in order to maintain passivity constraints.

3. This component will always provide attenuation; for example, either $Loss$=20 or $Loss$ =-20 will result in 20 dB attenuation.

4. When used in time domain simulations, set $ReturnPhase$=0; any other $ReturnPhase$ value will not produce proper results because $(+/-)j$ in the frequency domain corresponds to a non-causal impulse response in time domain.

5. When checking passivity, remember that infinitesimal numerical differences can make a difference. For example, an Attenuator with

$$|S21| = |S12| = 0.8$$
$$|S11| = |S22| = \sqrt{1 - |S21|^2} = 0.6$$
$$ReturnPhase <> 0$$

is theoretically passive. However, it may give a non-passivity warning in ADS due to infinitesimal numerical differences. Subtracting a small number, say 1e-12, from all S-parameters should address the problem without causing noticeable changes in the results.

6. *CheckPassivity* controls whether a separate passivity check is enabled. This is what outputs the *Linear S_Port ... is non-passive* warning. *ReturnPhase* controls whether limiting is enabled. This is what outputs the *VSWR limited ...* warning. When limiting is enabled, it implies a passivity check to find out what limiting, if any, should be done (refer to note 7). This passivity check is silent and never produces a non-passivity warning. When *ReturnPhase=0*, *VSWR* limiting is done before the passivity check if *CheckPassivity=yes*. This means that for *ReturnPhase=0*, a non-passivity warning will never be issued regardless of whether *CheckPassivity* is set to yes or no. For *CheckPassivity=no*, a passivity check will not be done; therefore a non-passivity warning will not be output. For *CheckPassivity=yes*, a passivity check will be done but the parameters have already been limited to ensure passivity so the component will be found to be passive and a non-passivity warning will not be output.

7. This note answers two questions regarding Attenuator:
   **Q** How does Attenuator do its passivity check and how/why does it perform limiting to ensure passivity?
   **A** The S-matrix for this attenuator is

$$S = \begin{bmatrix} S11 & S12 \\ S21 & S22 \end{bmatrix}$$

   We have

$$S = S12 = 10^{\frac{-abs(Loss)}{20}} = T$$

   and we see that $T$ is real and fulfils $0 \le T \le 1$ for *Loss* values from minus infinity to infinity. We also have

$$|S11| = |S22| = scale \times \frac{VSWR2 - 1}{VSWR2 + 1} = R$$

   where
   scale = if ( *ReturnPhase=0*) then 1 elseif (*ReturnPhase<0*) then -*j* else *j* endif
   *VSWR2* = if (*ReturnPhase=0*) then min( *VSWR* , *MaxVSWR* ) else *VSWR* endif
   *MaxVSWR*= (1+*MaxR*)/(1-*MaxR*)
   *MaxR* = maximum value of *R* which guarantees passivity, TBD
   and we see that *R* is real for *ReturnPhase=0* and fulfils $0 \le R \le 1$ for *VSWR* values from 1 to infinity. The S-matrix for *ReturnPhase=0* is therefore

$$S = \begin{bmatrix} R & T \\ T & R \end{bmatrix}$$

   The condition for passivity is that eigenvalues(transpose(conjugate(S)) × S) ≤ 1. For the real and symmetric matrix above, this reduces to eigenvalues( S × S ) ≤ 1. We have

$$S \times S = \begin{bmatrix} R^2 + T^2 & 2 \times R \times T \\ 2 \times R \times T & R^2 + T^2 \end{bmatrix}$$

   and find the eigenvalues $R^2 - 2 \times R \times T + T^2$ and $R^2 + 2 \times R \times T + T^2$. Letting these

eigenvalues equal to 1 (the max value for passivity), we get

$R^2 - 2 \times R \times T + T^2 = 1 \rightarrow R = -1 + T$ and $R = 1 + T$

$R^2 + 2 \times R \times T + T^2 = 1 \geq R = -1 - T$ and $R = 1 - T$

and see that eigenvalues ( $S \times S$ ) $\leq 1$ implies $R \geq -1 + T$, $R \leq 1 + T$, $R \geq -1 - 7$ and $R \leq 1 - T$. For $0 \leq R$, $T \leq 1$, the first three inequalities are always satisfied and the last one gives $MaxR = 1 - T$.

**Q** I have a dissipationless attenuator. It has a given $S21$ and because it is dissipationless I know that

$$|S11| = \sqrt{1 - |S21|^2}$$

.

For instance, I might have $S21 = 0.8$ and I know that $|S11| = 0.6$ corresponding to a $VSWR$ of 4. I enter $S21 = 0.8$ and $S11 = 0.6$ for the attenuator but when I simulate I'm told that $VSWR$ is limited to 1.5 corresponding to an $S11$ value of 0.2. I know that my attenuator is passive with $|S11| = 0.6$. Isn't the limiting by Attenuator too restrictive and wrong?

**A** Let's examine the above S-matrix. It is

$$\begin{bmatrix} 0.6 & 0.8 \\ 0.8 & 0.6 \end{bmatrix}$$

and it does fulfill

$$\sum_{i=1}^{2} (|Sij|^2) = 1 \quad [j = 1, 2]$$

However, for a dissipationless network it must also fulfill

$$\sum_{i=1}^{2} (Sij \times conjugate(Siq)) = 0 \quad [j = 1, 2; \ q = 1, 2; \ q < >j]$$

and it clearly does not. What is wrong is that $S11 = S22$ have zero phase. To remedy this, we could consider

$$\begin{bmatrix} j \times 0.6 & 0.8 \\ 0.8 & j \times 0.6 \end{bmatrix}$$

which fulfills both of the above equations. A similar matrix could be constructed by scaling with $-j$. These matrices must be entered into ADS with $ReturnPhase > 0$ and $ReturnPhase < 0$, respectively, not the default $ReturnPhase = 0$. For $ReturnPhase < >0$, no limiting is done on $VSWR$ and the above concern is void. An Attenuator with the original S-matrix is not dissipationless and must be limited in order to be passive.

All this generalizes to N × N matrices. Look, for example, at a standard four-port directional coupler. The elements are not all real. There are $j$'s floating around. Why? This is why.

A quick check shows that the eigenvalues of

$$\begin{bmatrix} 0.2 & 0.8 \\ 0.8 & 0.2 \end{bmatrix} \times \begin{bmatrix} 0.2 & 0.8 \\ 0.8 & 0.2 \end{bmatrix}$$

are 0.36 and 1.0 and that the eigenvalues of

$$\begin{bmatrix} 0.6 & 0.8 \\ 0.8 & 0.6 \end{bmatrix} \times \begin{bmatrix} 0.6 & 0.8 \\ 0.8 & 0.6 \end{bmatrix}$$

are 0.04 and 1.96 showing again that the circuit corresponding to the former/limited matrix is passive (largest eigenvalue of $S \times S$ equals one) while the circuit corresponding to the latter/unlimited matrix is not passive (largest eigenvalue of $S \times S$

> 1).

# Balun3Port (Balun, 3-port)

## Symbol



## Parameters

None

## Notes/Equations

1. Balun3Port realizes the ideal transformation between a balanced differential-mode signal and unbalanced, single-ended signals. It can be useful to connect a source to a differentially fed circuit, although it does ignore common-mode effects.

2. Balun3Port realizes the voltage and current transformations given by:

   $v_d = v_+ - v_-$

   $i_+ = -i_- = -i_d$

   where

   $v_d$ / $i_d$ = the differential mode voltage/current at pin d

   $v_+$ / $i_+$ = the single line voltage/current at pin +

   $v_-$ / $i_-$ = the single line voltage/current at pin -

   See *P.R. Gray and R.G. Meyer, "Analysis and Design of Analog Integrated Circuits", Third Edition, John Wiley & Sons, NY, 1993,* equations 3.91 & 3.92.
   The minus signs in the current definitions are due to the standard definition of currents directed *into* the Balun3Port component.

3. An equivalent functionality can be realized with a Balun4Port that has the common-mode pin grounded. However, the Balun3Port provides better convergence properties.

4. Balun3Port is bi-directional. When fed at the differential-mode pin, it realizes the transformations:

   $v_+ = -v_- = (v_d)/2$

   $i_+ = -i_- = -i_d$

5. Examples of using Balun3Port to convert between (unbalanced) ADS sources and balanced circuits can be found in the ADS examples directory; access these examples from the ADS Main window **File** > **Open** > **Example**.
   - *RFIC > MixerDiffMode_wrk* demonstrates the use of Balun4Port to present differential-mode sources (as well as common mode biases) to the RF and LO inputs. It also shows the use of Balun3Port to single-ended (differential-mode)

IF output, which is needed to properly calculate the noise figure.

- *BehavioralModels > DifferentialModels_wrk* demonstrates the use of Balun3Port and Balun4Port in conjunction with single-ended System-Data Models in order to create a data-based behavioral model of a differentially fed mixer.
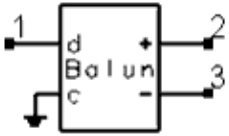
# Balun4Port (Balun, 4-port)

## Symbol
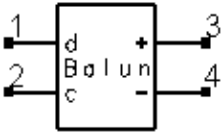


## Parameters

None

## Notes/Equations

1. Balun4Port realizes the ideal transformation between balanced (differential- and common-mode) signals and unbalanced (single-ended) signals. It can be used to connect sources to a differentially fed circuit, particularly when modeling common-mode effects are important.
2. Balun4Port realizes voltage and current transformations given by:

   $v_d = v_+ - v_-$

   $v_c = (v_+ + v_-)/2$

   $i_d = -(i_+ - i_-)/2$

   $i_c = -(i_+ + i_-)$

   where

   $v_d/i_d$ = differential mode voltage/current at pin d

   $v_c/i_c$ = common mode voltage/current at pin c

   $v_+/i_+$ = single line voltage/current at pin +

   $v_-/i_-$ = single line voltage/current at pin -

   See *P.R. Gray and R.G. Meyer, "Analysis and Design of Analog Integrated Circuits", Third Edition, John Wiley & Sons, NY, 1993,* equations 3.91 and 3.92.
   The minus signs in the current definitions are due to the standard definition of currents directed *into* the Balun4Port component.
3. Balun4Port is bi-directional. It converts common/differential-mode signals into two single-ended signals, as well as converting two single-ended signals into common/differential mode signals.
4. If common-mode effects are not desired, Balun3Port provides an equivalent, but numerically more robust, result as grounding the common-mode pin of Balun4Port.
5. Examples of using Balun4Port to convert between (unbalanced) ADS sources and balanced circuits can be found in the ADS examples directory; access these examples from the ADS Main window **File** > **Open** > **Example**.
   - *RFIC > MixerDiffMode_wrk* demonstrates the use of Balun4Port to present

differential-mode sources (as well as common mode biases) to the RF and LO inputs. It also shows the use of Balun3Port to single-ended (differential-mode) IF output, which is needed to properly calculate the noise figure.

- *BehavioralModels > DifferentialModels_wrk* demonstrates the use of Balun3Port and Balun4Port in conjunction with single-ended System-Data Models in order to create a data-based behavioral model of a differentially fed mixer.

# Balun6Port (Balun, 6-port)

## Symbol
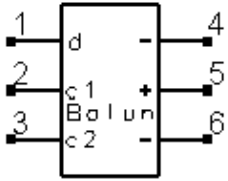


## Parameters

None

## Notes/Equations

1. Balun6Port is based on ideal transformers; it can be used to transform a single-ended signal to a differential signal with two ground planes such as a stripline.
2. This component passes DC.

# Circulator (Ideal 3-Port Circulator)

## Symbol



## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| F1 | First frequency breakpoint | | |
| F2 | Second frequency breakpoint | | |
| F3 | Third frequency breakpoint | | |
| Loss1 | Attenuation for frequencies ≤ F1 | dB | 0 |
| Loss2 | Attenuation for F1< frequency ≤ F2 | dB | 0 |
| Loss3 | Attenuation for F2 > frequency ≤ F3 | dB | 0 |
| Loss4 | Attenuation for frequencies > F3 | dB | 0 |
| VSWR1 | Voltage standing wave ratio at both ports for frequencies ≤ F1 | None | 1 |
| VSWR2 | Voltage standing wave ratio at both ports for frequencies > F1, ≤ F2 | None | 1 |
| VSWR3 | Voltage standing wave ratio at both ports for frequencies > F2, ≤ F3 | None | 1 |
| VSWR4 | Voltage standing wave ratio at both ports for frequencies > F3 | None | 1 |
| Isolat | Isolation | dB | 100 |
| Z1 | Reference impedance for port 1 | | |
| Z2 | Reference impedance for port 2 | | |
| Z3 | Reference impedance for port 3 | | |
| Temp | Temperature | °C | None |
| CheckPassivity | Check passivity flag: if set to yes, a passivity check is performed and a warning is output if the device is not passive; if set to no, a passivity check is not performed. | None | yes |

## Notes/Equations

1. Up to three frequency breakpoints can be used to define four bands. Each band will have a different Loss# and VSWR#. If no frequencies are specified, then this model is frequency independent and only uses Loss1 and VSWR1.
2. A loss is modeled regardless of the sign of the Loss parameter value.
3. All ports have the same VSWR.
4. Isolation is constant across all frequency bands.
5. The 3 × 3 S-matrix for Circulator is described by

$$S_{11} = S_{22} = S_{33} = \frac{VSWR - 1}{VSWR + 1}$$

$$S_{12} = S_{23} = S_{31} = 10^{(-abs(Isolat)/20)}$$
$$S_{13} = S_{21} = S_{32} = 10^{(-abs(Loss)/20)}$$

# CouplerDual (Dual Coupler)

## Symbol



## Parameters

| Name | Description | Units | Default |
|---|---|---|---|
| Coupling | Coupling factor | dB | 10 |
| F1 | First frequency breakpoint | | |
| F2 | Second frequency breakpoint | | |
| F3 | Third frequency breakpoint | | |
| MVSWR1 | Main arm Voltage standing wave ratio for frequency ≤ F1 | None | 1 |
| MVSWR2 | Main arm Voltage standing wave ratio for F1 < frequency ≤ F2 | None | 1 |
| MVSWR3 | Main arm voltage standing wave ratio for F2 < frequency ≤ F3 | None | 1 |
| MVSWR4 | Main arm voltage standing wave ratio for frequency > F3 | None | 1 |
| CVSWR1 | Coupled arm voltage standing wave ratio for frequency ≤ F1 | None | 1 |
| CVSWR2 | Coupled arm voltage standing wave ratio for F1 < frequency ≤ F2 | None | 1 |
| CVSWR3 | Coupled arm voltage standing wave ratio for F2 < frequency ≤ F3 | None | 1 |
| CVSWR4 | Coupled arm voltage standing wave ratio for frequency > F3 | None | 1 |
| Loss1 | Attenuation for frequency ≤ F1 | dB | 0 |
| Loss2 | Attenuation for F1 < frequency ≤ F2 | dB | 0 |
| Loss3 | Attenuation for F2 < frequency ≤ F3 | dB | 0 |
| Loss4 | Attenuation for frequency > F3 | dB | 0 |
| Direct1 | Directivity for frequency ≤ F1 | dB | 100 |
| Direct2 | Directivity for F1 < frequency ≤ F2 | dB | 100 |
| Direct3 | Directivity for F2 < frequency ≤ F3 | dB | 100 |
| Direct4 | Directivity for frequency > F3 | dB | 100 |
| ZRef | Reference impedance for all ports | Ohm | 50 |
| Temp | Temperature | °C | None |
| CheckPassivity | Check passivity flag: if set to yes, a passivity check is performed and a warning is output if the device is not passive; if set to no, a passivity check is not performed. | None | yes |

## Notes/Equations

1. Port 1 is the input port; port 2 is the through port; port 3 is the coupled port; port 4 is the isolated port. Note that the placement of these pins in ADS differs from many standard textbooks.

2. Both ports of the main arm (ports 1 and 2) are assumed to have the same VSWR. Both ports of the coupled arm (ports 3 and 4) are also assumed to have the same VSWR.
3. *Coupling* is assumed to be constant across all frequency bands.
4. *Loss* is the dissipation of the coupler; it affects S21 and S12 only. These S-parameters are calculated as if the coupler was dissipationless and are then scaled by $10^{(-abs(Loss)/20)}$
5. Internally, coupler isolation is calculated as *Coupling + Direct* in dB.
6. Up to three frequency breakpoints can be used to define four bands, with different directivity losses and VSWR values for each frequency band. If no frequencies are specified, then this model is frequency independent and uses Direct1, Loss1, and VSWR1 only.
7. There is a 90-degree phase shift between the input and the coupled ports.
8. In the coupled arm, the two ports are uncoupled.
9. This is an ideal component with potentially non-causal behavior. It is not recommended for transient simulation.

# CouplerSingle (Single Coupler)

**Symbol**



**Parameters**

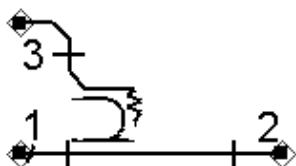| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Coupling | Coupling factor | dB | 10 |
| F1 | First frequency breakpoint | | |
| F2 | Second frequency breakpoint | | |
| F3 | Third frequency breakpoint | | |
| MVSWR1 | Main arm Voltage standing wave ratio for frequency ≤ F1 | None | 1 |
| MVSWR2 | Main arm Voltage standing wave ratio for F1 < frequency ≤ F2 | None | 1 |
| MVSWR3 | Main arm voltage standing wave ratio for F2 < frequency ≤ F3 | None | 1 |
| MVSWR4 | Main arm voltage standing wave ratio for frequency > F3 | None | 1 |
| CVSWR1 | Coupled arm voltage standing wave ratio for frequency ≤ F1 | None | 1 |
| CVSWR2 | Coupled arm voltage standing wave ratio for F1 < frequency ≤ F2 | None | 1 |
| CVSWR3 | Coupled arm voltage standing wave ratio for F2 < frequency ≤ F3 | None | 1 |
| CVSWR4 | Coupled arm voltage standing wave ratio for frequency > F3 | None | 1 |
| Loss1 | Attenuation for frequencies ≤ F1 | dB | 0 |
| Loss2 | Attenuation for F1< frequency ≤ F2 | dB | 0 |
| Loss3 | Attenuation for F2 > frequency ≤ F3 | dB | 0 |
| Loss4 | Attenuation for frequencies > F3 | dB | 0 |
| Direct1 | Directivity for frequency ≤ F1 | dB | 100 |
| Direct2 | Directivity for F1 < frequency ≤ F2 | dB | 100 |
| Direct3 | Directivity for F2 < frequency ≤ F3 | dB | 100 |
| Direct4 | Directivity for frequency > F3 | dB | 100 |
| ZRef | Reference impedance for all ports | Ohm | 50 |
| Temp | Temperature | °C | None |
| CheckPassivity | Check passivity flag: if set to yes, a passivity check is performed and a warning is output if the device is not passive; if set to no, a passivity check is not performed. | None | yes |

**Notes/Equations**

1. Port 1 is the input port; port 2 is the through port; port 3 is the coupled port. Note that the placement of these pins in ADS differs from many standard textbooks.

2. Both ports of the main arm (ports 1 and 2) are assumed to have the same VSWR.
3. *Coupling* is assumed to be constant across all frequency bands.
4. *Loss* is the dissipation of the coupler; it affects S21 and S12 only. These S-parameters are calculated as if the coupler was dissipationless and are then scaled by $10^{(-abs(Loss)/20)}$

5. Internally, coupler isolation is calculated as *Coupling + Direct* in dB.
6. Up to three frequency breakpoints can be used to define four bands, with different directivity losses and VSWR values for each frequency band. If no frequencies are specified, then this model is frequency independent and uses Direct1, Loss1, and VSWR1 only.
7. There is a 90-degree phase shift between the input and the coupled ports.
8. In the coupled arm, the two ports are uncoupled.
9. This is an ideal component with potentially non-causal behavior. It is not recommended for transient simulation.

# Gyrator (Gyrator)

**Symbol**
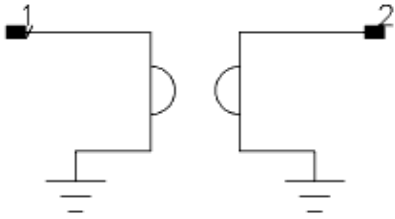


**Illustration**
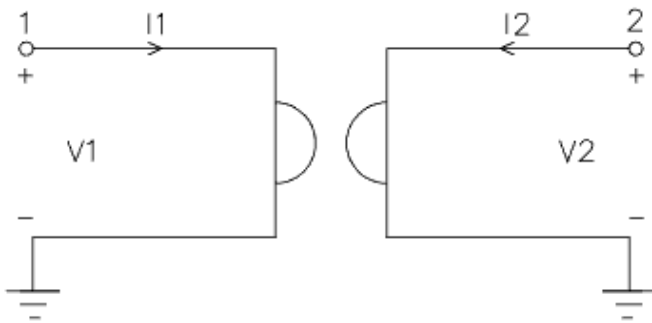


**Parameters**

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Ratio | Gyrator ratio | None | 1 |

**Notes/Equations**

1. $V1 = \text{Ratio} \times I2$
   $V2 = -\text{Ratio} \times I1$

$$S_{11} = S_{22} = \frac{r^2 - 1}{r^2 + 1}$$

$$S_{21} = -S_{12} = \frac{-2r}{r^2 + 1}$$

$$r = \frac{Ratio}{Z_0}$$

# Hybrid90 (Ideal 90-degree Hybrid Coupler)

## Symbol



## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Loss | Insertion loss | dB | 0 |
| GainBal | Gain balance between output ports | dB | 0 |
| PhaseBal | Phase balance between output ports | deg | 0 |
| Rref | Port reference impedance | Ohm | 50 |
| Temp | Temperature | °C | None |
| CheckPassivity | Check passivity flag: if set to yes, a passivity check is performed and a warning is output if the device is not passive; if set to no, a passivity check is not performed. | None | yes |
| Delay | Time delay for output ports | sec | 1.0e-16 |

## Range of Usage

Loss $\geq$ 0 dB
Delay $\geq$ 1.0e-16 sec

## Notes/Equations

1. Although hybrid couplers are typically designed to give an equal power split between the coupled and the direct ports, the choice of coupling coefficients has been provided here for greater flexibility where a coupling of other than 3 dB is required.
2. The loss term is applied to both the coupling and the direct transmission coefficients.
3. Pin designations (directional coupler notation/0-90 notation):
   (IN) = input port #1
   (IS0) = isolated output/input port #2
   (0) = coupled output / in-phase output
   (−90) = direct output/90-degree phase output

$$|S_{0,IN}|(dB) = -10 \times \log(1 + 10^{(-GainBal/10)}) - Loss$$

$$|S_{-90,IN}|(dB) = -10 \times \log(1 + 10^{(-GainBal/10)}) - Loss - GainBal$$

$$|S_{0,ISO}|(dB) = -10 \times \log(1 + 10^{(-GainBal/10)}) - Loss - GainBal$$

$$|S_{-90,ISO}|(dB) = -10 \times \log(1 + 10^{(-GainBal/10)}) - Loss$$

$$phase(S_{0,IN}) = -360 \times Delay \times frequency(degrees)$$

$$phase(S_{-90,IN}) = -360 \times Delay \times frequency - 90 - PhaseBal(degrees)$$

$$phase(S_{0,ISO}) = -360 \times Delay \times frequency - 90 - PhaseBal(degrees)$$

$$phase(S_{-90,ISO}) = -360 \times Delay \times frequency(degrees)$$

4. In general, Hybrid90 is not recommended for time-domain simulation. For the typical case of PhaseBal=0, time domain simulation will not produce proper results for $S_{90,IN}$ , or $S_{0,ISO}$ because ( )j in the frequency domain corresponds to a non-causal impulse response in time domain.

5. This component has no default artwork associated with it.

# Hybrid180 (Ideal 180-degree Hybrid Coupler)

## Symbol



## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Loss | Insertion loss | dB | 0 |
| GainBal | Gain balance between output ports | dB | 0 |
| PhaseBal | Phase balance between output ports | deg | 0 |
| Rref | Port reference impedance | Ohm | 50 |
| Temp | Temperature | °C | None |
| CheckPassivity | Check passivity flag: if set to yes, a passivity check is performed and a warning is output if the device is not passive; if set to no, a passivity check is not performed. | None | yes |
| Delay | Time delay for output ports | sec | 1.0e-16 |

## Range of Usage

Loss ≥ 0 dB
Delay ≥ 1.0e-16 sec

## Notes/Equations

1. Although hybrid couplers are typically designed to give an equal power split between the coupled and the direct ports, the choice of coupling coefficient has been provided here for greater flexibility where a coupling of other than 3 dB is required.
2. The loss term is applied to both the coupling and the direct transmission coefficients.
3. Pin designations (directional coupler notation/sum-difference notation):
   (IN) = input port/input port#1
   (IS0) = isolated output/input port #2
   ( Σ) = coupled output/summation output
   (Δ) = direct output/difference output

$$|S_{\Sigma, IN}|(dB) = -10 \times \log(1 + 10^{(-GainBal/10)}) - Loss - GainBal$$

$$|S_{\Delta, IN}|(dB) = -10 \times \log(1 + 10^{(-GainBal/10)}) - Loss$$

$$|S_{\Sigma, ISO}|(dB) = -10 \times \log(1 + 10^{(-GainBal/10)}) - Loss$$

$$|S_{\Delta, ISO}|(dB) = -10 \times \log(1 + 10^{(-GainBal/10)}) - Loss - GainBal$$

$$phase(S_{\Sigma, IN}) = -360 \times Delay \times frequency - PhaseBal(degrees)$$

$$phase(S_{\Delta, IN}) = -360 \times Delay \times frequency(degrees)$$

$$phase(S_{\Sigma, ISO}) = -360 \times Delay \times frequency(degrees)$$

$$phase(S_{\Delta, ISO}) = -360 \times Delay \times frequency - PhaseBal + 180(degrees)$$

4. When used in time domain simulation, set PhaseBal=0; for any PhaseBal value other than n × 180, time domain simulation will not produce proper results because the frequency domain specification corresponds to a non-causal impulse response in time domain.

5. This component has no default artwork associated with it.

# IsolatorSML (SMLIsolator)

## Symbol

## Parameters

| Name | Description | Units | Default |
|---|---|---|---|
| F1 | First frequency breakpoint | | |
| F2 | Second frequency breakpoint | | |
| F3 | Third frequency breakpoint | | |
| Loss1 | Attenuation for frequencies ≤ F1 | dB | 0 |
| Loss2 | Attenuation for F1< frequency ≤ F2 | dB | 0 |
| Loss3 | Attenuation for F2 > frequency ≤ F3 | dB | 0 |
| Loss4 | Attenuation for frequencies > F3 | dB | 0 |
| VSWR1 | Voltage standing wave ratio at both ports for frequencies ≤ F1 | None | 1 |
| VSWR2 | Voltage standing wave ratio at both ports for frequencies > F1, ≤ F2 | None | 1 |
| VSWR3 | Voltage standing wave ratio at both ports for frequencies > F2, ≤ F3 | None | 1 |
| VSWR4 | Voltage standing wave ratio at both ports for frequencies > F3 | None | 1 |
| Isolat | Isolation | dB | 100 |
| Z1 | Reference impedance for port 1 | | |
| Z2 | Reference impedance for port 2 | | |
| Temp | Temperature | °C | None |
| CheckPassivity | Check passivity flag: if set to yes, a passivity check is performed and a warning is output if the device is not passive; if set to no, a passivity check is not performed. | None | yes |

## Notes/Equations

1. All ports are assumed to have the same VSWR.
2. Isolation is assumed to be constant across all frequency bands.
3. Up to three frequency breakpoints can be used to define four bands, with different losses and VSWR for each frequency band. If frequencies are not specified, this model is frequency independent and uses Loss1 and VSWR1 only.

# LOS_Link (Line-Of-Sight Antenna Link)

## Symbol



## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| CenterFreq | Link center frequency | GHz | 1 |
| BW | Link bandwidth | MHz | 100 |
| TxGain | Transmitting antenna gain | dB | 0 |
| TxVSWR | Transmitting antenna VSWR | None | 1 |
| TxParabolaD | Transmitting parabolic antenna diameter; overwrites TxGain | | |
| TxEfficiency | Transmitting parabolic antenna efficiency; overwrites TxGain | None | None |
| RxGain | Receiving antenna gain | dB | 0 |
| RxVSWR | Receiving antenna VSWR | None | 1 |
| RxParabolaD | Receiving parabolic antenna diameter; overwrites RxGain | | |
| RxEfficiency | Receiving parabolic antenna efficiency; overwrites RxGain | None | None |
| RxNoiseTemp | Receiving antenna noise temperature | K | 150 |
| PathLength | Line-of-sight path length | km | 10 |
| NotchFreq | Notch frequency due to ground reflection path interference | | |
| NotchDepth | Power of reflected ray relative to direct ray | dB | None |
| DeltaDelay | Time delay of ground reflection path w.r.t. LOS path | | |
| Z1 | Transmitting antenna reference impedance | Ohm | 50 |
| Z2 | Transmitting antenna reference impedance | Ohm | 50 |

## Range of Usage

PathLength > 10 wavelengths

## Notes/Equations

1. Pathloss is infinite outside a window defined by BW.
2. Transmitting antenna gain can be defined by TxGain or by TxParabolaD and TxEfficiency. If all three parameters are specified, TxParabolaD and TxEfficiency will overwrite TxGain. The same applies to receiving antenna as well.
3. The S-Parameter and noise implementation of this model is as follows. S11, S22, S12, S21, and I_Noise are derived from the user-specified LOS_link parameters.
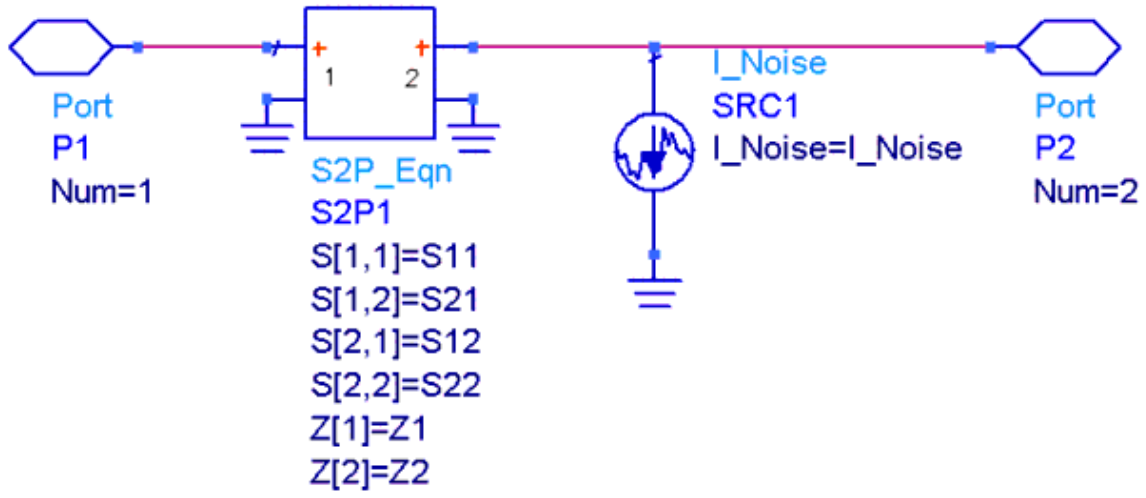
# Pad (Pi or Tee Format)

## Symbol
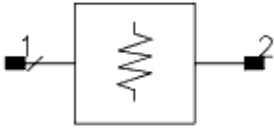


## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| NetType | Network type: Pi or Tee | None | Pi |
| Loss | Attenuation | dB | 0 |
| R1 | Reference resistance for port 1 | Ohm | 50 |
| R2 | Reference resistance for port 2 | Ohm | 50 |
| Temp | Temperature | °C | None |
| CheckPassivity | Check passivity flag: if set to yes, a passivity check is performed and a warning is output if the device is not passive; if set to no, a passivity check is not performed. | None | yes |

## Range of Usage

R1 > 0
R2 > 0

## Notes/Equations

1. This 2-port component provides a:
   - Match to resistance $R_1$ at port 1 when port 2 is terminated in $R_2$
   - Match to $R_2$ at port 2 when port 1 is terminated in $R_1$
   - Desired attenuation, DB
2. Resistive networks can achieve simultaneous match at input and output if the insertion loss factor A exceeds a minimum value given by

$$A_{min} = 2 \times \frac{R_1}{R_2} \times \left(1 + \sqrt{1 - \frac{R_2}{R_1}}\right) - 1.0 \quad R_1 > R_2$$

$$2 \times \frac{R_2}{R_1} \times \left(1 + \sqrt{1 - \frac{R_1}{R_2}}\right) - 1.0 \quad R_2 > R_1$$

   where:
   $DB_{MIN} = 10 \times \log_{10}(A_{MIN})$
3. The values shown in the Pi network equivalent circuit are given by:

$$G_c = 2 \times \frac{\sqrt{A}}{(A-1) \times \sqrt{R_1 \times R_2}}$$

$$G_a = \frac{A+1}{(A-1) \times Z_1} - G_c$$

$$G_b = \frac{A+1}{(A-1) \times R_2} - G_c$$

The values shown in the Tee network circuits are given by

$$R_c = \frac{2 \times \sqrt{A} \times \sqrt{R_1 \times R_2}}{(A-1)}$$

$$R_a = \frac{R_1 \times (A+1)}{A-1} - R_c$$

$$R_b = \frac{R_2 \times (A+1)}{A-1} - R_c$$

## Equivalent Circuit



Pi Network
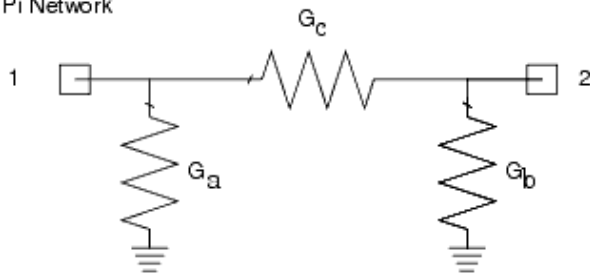
Tee Network
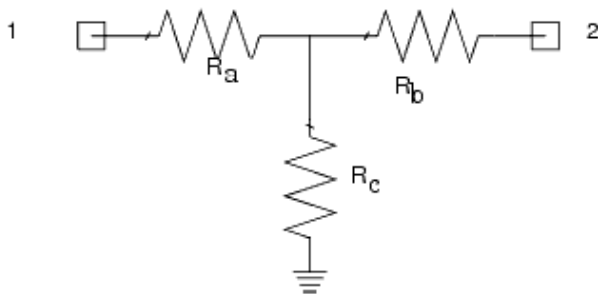
# PhaseShiftSML (Phase Shifter)

## Symbol
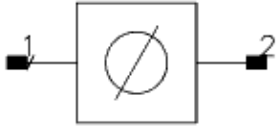
## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Phase | Phase shift | deg | 90 |
| PhaseSlope | Phase slope per frequency octave | None | None |
| FreqStart | Frequency where slope begins | | |
| RTConj | Apply reverse transmission conjugate: yes or no | None | no |
| ZRef | Reference impedance for all ports | Ohm | 50 |

## Range of Usage

FreqStart ≥ 0

## Notes/Equations

1. The output frequency spectrum is equal to the input spectrum shifted by the specified phase.

2. $\theta$ (f) = Phase (for freq < FreqStart)
   $\theta$ (f) = Phase + PhaseSlope × $\text{LOG}_2$ (freq / FreqStart) (for freq ≥ FreqStart)

   if FreqStart = 0, then phase slope is 0 regardless of the PhaseSlope setting
   where freq = simulation frequency

3. $$\frac{V2}{V1} = \frac{I2}{I1} = e^{j\theta(f)}$$
   $$S_{11} = S_{22} = 0$$
   $$S_{21} = e^{j\theta(f)}$$

   if RTConj = NO, $S_{12} = S_{21}$

   if RTConj = YES, $S_{12} = S_{21}$*

4. The S-parameter implementation for the phase shifter is shown here:
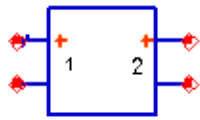
S2P_Eqn
S2P1
S[1,1]=0
S[1,2]=S12
S[2,1]=S21
S[2,2]=0
Z[1]=ZRef
Z[2]=ZRef

5. This component is intended for use with RF simulators, such as S-parameter, AC, HB, and in Circuit Envelope. It is not recommended for use in transient or baseband envelope simulations.

# PwrSplit2 (2-Way Power Splitter)
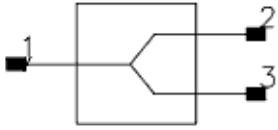
## Symbol



## Parameters

| Name | Description | Units | Default |
|---|---|---|---|
| S21 | Transmission Coefficient, port1 to port2, complex number: 1 + j × 0, polar(), dbpolar() | None | 0.707 |
| S31 | Transmission coefficient, port1 to port3, complex number: 1 + j × 0, polar(), dbpolar() | None | 0.707 |
| S11 | Port1 reflection coefficient, complex number: 1 + j × 0, polar(), dbpolar() | None | 0 |
| S22 | Port2 reflection coefficient, complex number: 1 + j × 0, polar(), dbpolar() | None | 0 |
| Isolation | Isolation between port 2 and port 3 | dB | 100 |
| ZRef | Reference impedance for all ports | Ohm | 50 |
| Temp | Temperature | °C | None |
| CheckPassivity | Check passivity flag: if set to yes, a passivity check is performed and a warning is output if the device is not passive; if set to no, a passivity check is not performed. | None | yes |
| Delay | Time delay | | |

## Notes/Equations

1. $S_{12} = S_{21}$ , $S_{13} = S_{31}$ , $S_{33} = S_{22}$

2. PwrSplit2 can also be used as a power combiner.

3. Use the functions polar( ), dbpolar( ) to represent the S-parameters in terms of magnitude, phase or dB, phase.

# PwrSplit3 (3-Way Power Splitter)
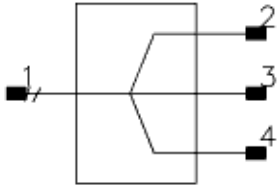
## Symbol



## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| S21 | Transmission coefficient, port1 to port2, complex number: $1 + j \times 0$, polar(), dbpolar() | None | 0.577 |
| S31 | Transmission coefficient, port1 to port3, complex number: $1 + j \times 0$, polar(), dbpolar() | None | 0.577 |
| S41 | Transmission coefficient, port1 to port4, complex number: $1 + j \times 0$, polar(), dbpolar() | None | 0.577 |
| ZRef | Reference impedance for all ports | Ohm | 50 |
| Temp | Temperature | °C | None |
| CheckPassivity | Check passivity flag: if set to yes, a passivity check is performed and a warning is output if the device is not passive; if set to no, a passivity check is not performed. | None | yes |
| Delay | Time delay | | |

## Notes/Equations

1. Ideal isolation exists for $S_{23}$ , $S_{24}$ , $S_{32}$ , $S_{34}$ , $S_{42}$ , and $S_{43}$ , that is,

   $S_{23} = S_{32} = S_{24} = S_{42} = S_{34} = S_{43} = 0$.

2. Ideal match exists for $S_{11}$ , $S_{22}$ , $S_{33}$ , and $S_{44}$ , that is,

   $S_{11} = S_{22} = S_{33} = S_{44} = 0$.

3. $S_{12} = S_{21}$ , $S_{13} = S_{31}$ and $S_{14} = S_{41}$.

4. PwrSplit3 can also be used as a power combiner.

5. Use the functions polar( ), dBpolar( ) to represent the S-parameters in terms of magnitude, phase or dB, phase.

# TimeDelay (Time Delay)

## Symbol
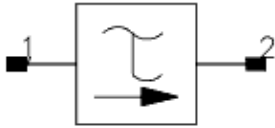


## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Delay | Time delay | sec | 1.0e-6 |
| RTConj | Apply reverse transmission conjugate: NO, YES | None | no |
| ZRef | Reference impedance for all ports | Ohm | 50 |

## Range of Usage

Delay ≥ 0

## Notes/Equations

1. The input frequency spectrum has a linear phase shift applied resulting in the output time waveform being a time shifted replication of the input waveform.

2. $S_{21} = e^{-j\omega Delay}$

   $S_{11} = S_{22} = 0$

   $\omega = 2\pi f$
   if RTConj = NO, $S_{12} = S_{21}$

   if RTConj = YES, $S_{12} = S_{21}{}^{*}$

3. Excessive values for the Delay parameter require large memory for Circuit Envelope and Transient simulations. For Circuit Envelope simulations, a safeguard limits any excessive delay to one-half of the stop time.

# Transformer (Ideal 4-Port Transformer)

## Symbol



## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| N | Turns ratio | | |
| ZRef | Reference impedance for all ports | | |

## Range of Usage

$N \neq 0$

## Notes/Equations

> **ℹ Note**
> This component is obsolete for new designs. It is available only for compatibility with designs created with 2002C or earlier releases.
>
> For new designs, use the TF transformer (Lumped-Components library).

1. Input pins are 1 and 3; output pins are 2 and 4.
2. The S-parameters of the component are determined as follows:

   $S_{11} = S_{24} = S_{33} = S_{42} = N^2 / (1 + N^2)$

   $S_{14} = S_{23} = S_{32} = S_{41} = -N / (1 + N^2)$

   $S_{12} = S_{21} = S_{34} = S_{43} = N / (1 + N^2)$

   $S_{13} = S_{22} = S_{31} = S_{44} = 1 / (1 + N^2)$

3. The input resistance $R_{IN}$ output resistance $R_{OUT}$ are related to N by:

   $$\sqrt{\frac{R_{IN}}{R_{OUT}}} = N$$

4. This is a noiseless component.
5. Because it is an ideal transformer, the impedance transformation is the same at DC as it is at nonzero frequencies.
6. This component passes DC.

# TransformerG (Transformer with Ground Reference)

## Symbol
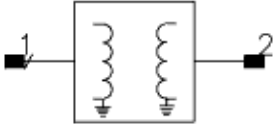


## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| N | Turns ratio | | |
| ZRef | Reference impedance for all ports | | |

## Range of Usage

N ≠ 0

## Notes/Equations

> **Note** This component is obsolete for new designs. It is available only for compatibility with designs created with 2002C or earlier releases.)
>
> For new designs use the TF transformer (Lumped-Components library) .

1. The input resistance $R_{IN}$ and $R_{OUT}$ are related to N by:

$$\sqrt{\frac{R_{IN}}{R_{OUT}}} = N$$

# TwoPort (2-Port Model)

## Symbol



## Parameters

| Name | Description | Units | Default |
|---|---|---|---|
| S21 | Forward transmission coefficient, complex number: 1 + j × 0, polar(), dbpolar() | None | 1 |
| S12 | Reverse transmission coefficient, complex number: 1 + j × 0, polar(), dbpolar() | None | 1 |
| S11 | Port1 reflection coefficient, complex number: 1 + j × 0, polar(), dbpolar() | None | 0 |
| S22 | Port2 reflection coefficient, complex number: 1 + j × 0, polar(), dbpolar() | None | 0 |
| ZRef | Reference impedance for all ports | Ohm | 50 |
| Temp | Temperature | °C | |
| CheckPassivity | Check passivity flag: if set to yes, a passivity check is performed and a warning is output if the device is not passive; if set to no, a passivity check is not performed. | None | yes |
| Delay | Time delay | | |

## Notes/Equations

1. Use the functions polar( ), dBpolar( ) to represent the S-parameters in terms of magnitude, phase or dB, phase.
2. The block parameters must be passive for the noise performance to be as expected.

# Phase Lock Loop Components

- *DivideByN (Divide by N)* (ccsys)
- *PhaseFreqDet2 (Frequency Detector, Baseband)* (ccsys)
- *PhaseFreqDet (Frequency Detector, Baseband)* (ccsys)
- *PhaseFreqDetCP (Frequency Detector, Baseband with Charge Pump)* (ccsys)
- *PhaseFreqDetTuned (Phase Frequency Detector, Tuned)* (ccsys)
- *PhaseNoiseMod (Phase Noise Modulator)* (ccsys)
- *VCO (Voltage Controlled Oscillator)* (ccsys)
- *VCO DivideByN (VCO Divide By N)* (ccsys)

# DivideByN (Divide by N)

## Symbol
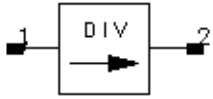


## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| FnomIn | Nominal input frequency | kHz | 1.0 |
| N | Divide number | None | 11 |

## Notes/Equations

1. This model performs a divide-by-N function on either a baseband-input or a selected-carrier input frequency. The model operates in transient, harmonic balance, or circuit envelope simulation.
   In transient, all signals are considered baseband.
   In circuit envelope, the FnomIn parameter defines which analysis frequency to use. If the analysis frequency is not within 0.5/timestep, a warning is issued and 0 Hz will be used for the analysis frequency.

2. The input impedance is infinite. The output impedance fixed at 1 Ohm. If the output is in baseband mode, then its open circuit voltage is equal to the phase of the divided signal in radians. This results in a sawtooth waveform for a CW input. A sawtooth is output (instead of a square wave) to allow for the calculation of the actual frequency from the slope and for the accurate detection of zero crossings without the sampling jitter that would be introduced by the fixed rate sampling of a square wave. If the output is not in baseband mode, then the open circuit voltage is a 1V complex sinusoid at the divided frequency. Whether or not the output is in baseband is determined, in Circuit Envelope mode, by determining the carrier frequency, possibly dc, that is closest to FnomIn; if none are close enough, a warning is issued.

3. In non-baseband mode of operation, the divider works by directly extracting the phase of the complex input. The delay of DivideByN = max(timestep, N/(2 × Fin), where Fin is the actual analysis frequency corresponding to FnomIn.
   In baseband mode of operation, the period of the input is determined by detecting when the baseband signal rises through the 0.5V threshold. From this period information, the phase of the divided signal is calculated. For a more standard counter model, see the counter model under probes.

4. Divider with an RF Carrier Input show examples of the divider operating with an RF carrier input: one divider outputs a baseband output; one divider outputs the divided frequency at the other carrier frequency. Note that N can be time varying in order to simulate the effect of a fractional-N divider. Output Waveforms shows the simulation results.
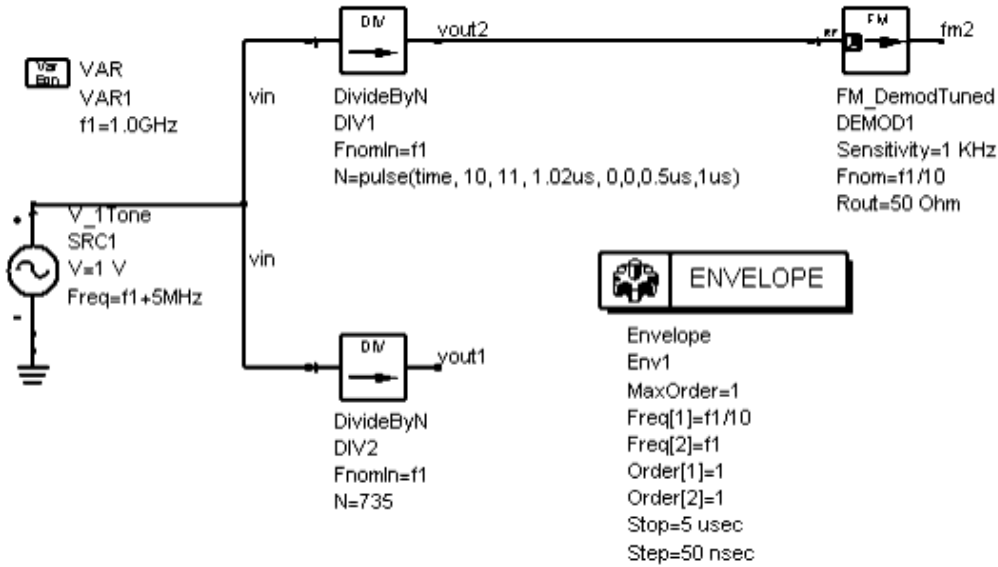
## Divider with an RF Carrier Input
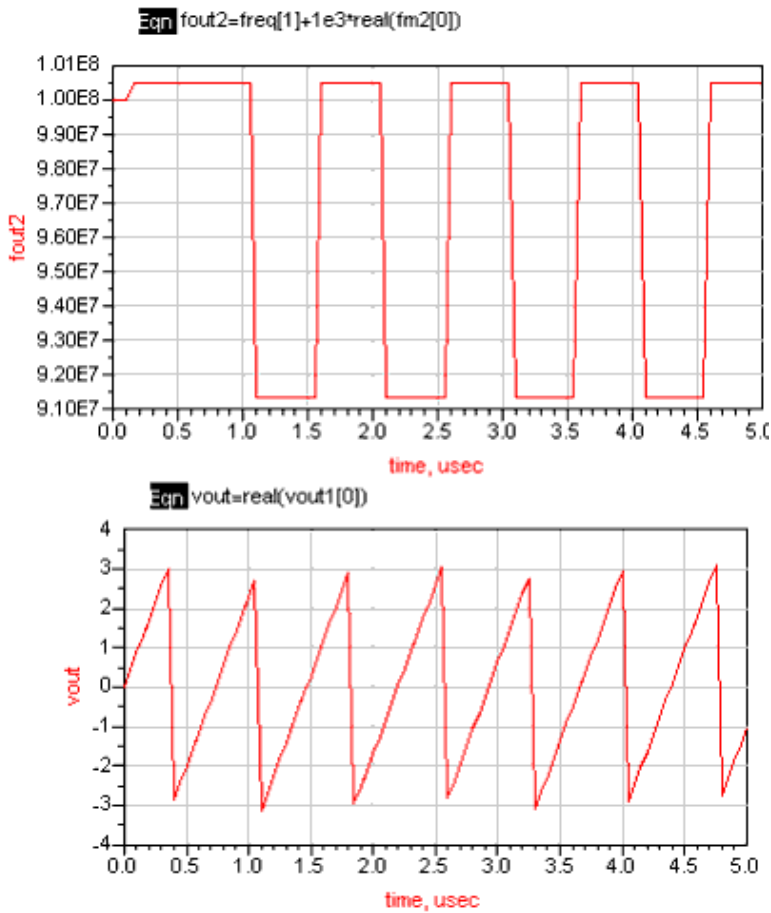


## Output Waveforms

# PhaseFreqDet2 (Frequency Detector, Baseband)

**Symbol**
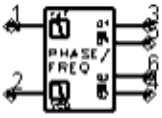


**Parameters**

| Name | Description | Units | Default |
|---|---|---|---|
| Vhigh | High-state output voltage | V | 5 |
| Vlow | Low-state output voltage | V | 0 |
| DeadTime | Dead zone pulse width | psec | 0 |
| Jitter | Input time jitter | psec | 0 |

**Notes/Equations**

1.  This baseband phase-frequency demodulator is used in transient or circuit envelope simulation. It models the digital behavior of common D flip-flop type phase-frequency detectors often used in phase-locked loops. The four outputs are ideal, zero impedance voltage sources. The two inputs have infinite impedance, and only the baseband portion of the two input voltages are used to determine threshold-crossing timing.

2.  The following FDD modeling equations illustrate how the shape of the four outputs relate to the input phase difference.
    The phase difference between the input on port 1 and port 2 is calculated as ns=_phase_freq(1,2), and the quantities n1= (Vhigh-Vlow) x real(ns) and n2= (Vhigh-Vlow) x imag(ns) are introduced.
    Outputs on ports 3-6 are then
    port 3: Q1=Vlow + n1
    port 4: Q2_bar = Vhigh - n2
    port 5: Q1_bar = Vhigh - n1 (only PhaseFreqDet2, not PhaseFreqDet)
    port 6: Q2 = Vlow + n2 (only PhaseFreqDet2, not PhaseFreqDet)
    with the following trigger events:
    Trig[1] = _xcross (1,0.5,1)
    Trig[2] = _xcross (3,0.5,1)
    For details about these functions, refer to the FDD device documentation in the *Circuit Components Nonlinear Devices* manual.
    Note that you can push into the component to see the implementation of the component.

3.  As opposed to the tuned phase-frequency detector model, this model's output includes reference clock feed-through effects.

> ℹ️ **Note** The output of this model is a pulse train whose average value is proportional to the input phase difference, and may contain significant signal energy at the reference clock rate, and at clock harmonics. Typically, these must be filtered out before driving a VCO in a PLL application. The tuned phase-frequency detector output signal includes instantaneous phase difference information only (it does not contain reference frequency or harmonic content).

However, the penalty for this is that the timestep must be less than one-half the reference period, and typically less than one-tenth the period. To avoid the large amount of time jitter and phase noise that would normally be introduced by sampling at even these rates, the four digital outputs are also amplitude modulated to reflect the portion of a simulation timestep where the actual outputs would be high or low. For example, if based on the threshold-crossing timing (the pulse width should be 10 nsec, but the simulation timestep is 1µ), then the output amplitude for that timestep would only be 1% of the Vhigh level. While this will not properly model all the higher harmonics of the reference feed-through, it does accurately model both the dc term and the first few harmonics, and the corresponding reference sidebands of the VCO. As the analysis timestep is further reduced, the behavior becomes more truly digital in nature and more harmonics are effectively being simulated at the cost of slower simulations.

4. The trigger times for both inputs is determined by detecting when the baseband voltage rises through the 0.5V threshold. Linear interpolation is used to get much finer time resolution than the analysis timestep. To further reduce excessive sampling jitter, inputs to this detector can be sawtooth waveforms. While the detector will work reasonably well with sinusoidal inputs, given a small enough timestep, the timing jitter can be eliminated if the interpolation is done on the positive slope of a sawtooth waveform. This is the reason why the divide-by-N models output a sawtooth waveform when they operate in the baseband mode. Square wave-inputs should generally be avoided, because this will usually introduce significant timing jitter and phase noise into the simulation.

5. DeadTime specifies the period of time centered around the 0 phase output during which no output is generated from either the high or the low charge pump. Outside of this period, the output returns to the ideal pulse widths and amplitudes determined by the trigger crossings of the inputs and the charge pump currents.

6. The Jitter value defines the RMS time jitter associated with a trigger crossing on either input. The distribution is Gaussian and the noise spectrum is assumed flat out to the reference sampling frequency.

7. Other effects, such as asymmetry between the four different outputs, can be incorporated by changing the external components. One application of detector shows one application of this detector and Resultant output waveforms of detector the resultant output waveforms.
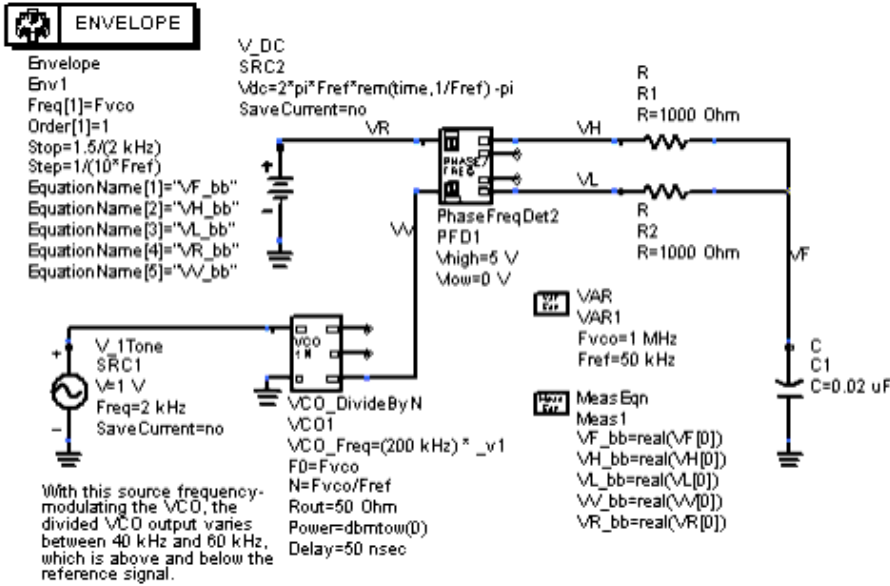
**One application of detector**
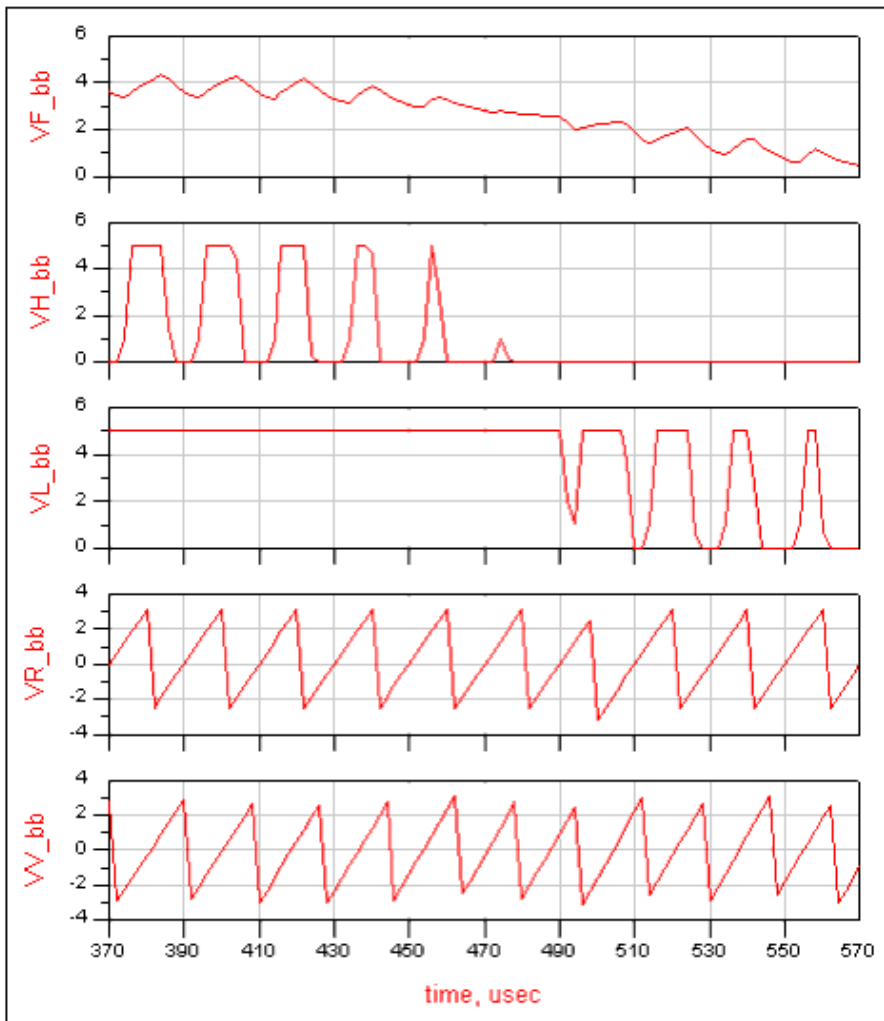
**Resultant output waveforms of detector**

# PhaseFreqDet (Frequency Detector, Baseband)

## Symbol
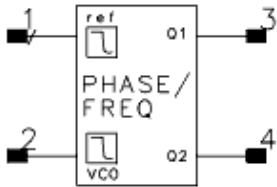


## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Vhigh | High-state output voltage | | |
| Vlow | Low-state output voltage | | |

## Notes/Equations

1. PhaseFreqDet is not selectable from the component palette or component library browser; to place this component, type its exact name into the Component History box above the drawing area, then move your cursor to the drawing area.
2. This baseband phase-frequency demodulator is used in transient or circuit envelope simulation. It models the digital behavior of common D flip-flop type phase-frequency detectors often used in phase-locked loops. The two outputs are ideal, zero impedance voltage sources; to model a pulsed current-source output, two VDCS-dependent sources must be added to the output of this model. The two inputs have infinite impedance, and only the baseband portion of the two input voltages are used to determine threshold-crossing timing.
3. As opposed to the tuned phase-frequency detector model, this model's output includes the effect of reference clock feed-through.

> **ℹ Note** The output of this model is a pulse train whose average value is proportional to the input phase difference, and may contain significant signal energy at the reference clock rate and at clock harmonics. These must be filtered out, typically before driving a VCO in a PLL application. The tuned phase-frequency detector output signal includes the instantaneous phase difference information only; it does not contain reference frequency or harmonic content.

However, the penalty for this is that the timestep must be less than one-half the reference period, and typically less than one-tenth the period. To avoid the large amount of time jitter and phase noise that would normally be introduced by sampling at even these rates, the two digital outputs are also amplitude modulated to reflect the portion of a simulation timestep where the actual outputs would be high or low. For example, if based on the threshold-crossing timing (the pulse width should be 10 nsec, but the simulation timestep is 1μ), then the output amplitude for that timestep would only be 1% of the Vhigh level. While this will not properly model all the higher harmonics of the reference feed-through, it does accurately model both the dc term and the first few harmonics, and the corresponding reference sidebands of the VCO. As the analysis timestep is further reduced, the behavior becomes more truly digital

in nature and more harmonics are effectively being simulated at the cost of slower simulations.

4. The trigger times for both inputs is determined by detecting when the baseband voltage rises through the 0.5V threshold. Linear interpolation is used to get much finer time resolution than the analysis timestep. To further reduce excessive sampling jitter, the inputs to this detector can be sawtooth waveforms. While the detector will work reasonably well with sinusoidal inputs, given a small enough timestep, timing jitter can be eliminated if the interpolation is done on the positive slope of a sawtooth waveform. This is the reason why the divide-by-N models output a sawtooth waveform when they operate in the baseband mode. Square-wave inputs should generally be avoided, because this will usually introduce significant timing jitter and phase noise into the simulation.

5. This model does not include any effects due to the finite duration of the flip-flop reset pulse and resultant zero-phase dead zone. Other effects, such as asymmetry between the two different outputs, can be incorporated by changing the external components. PhaseFreqDet Example shows an example using this detector; Output Waveforms shows the output waveforms.
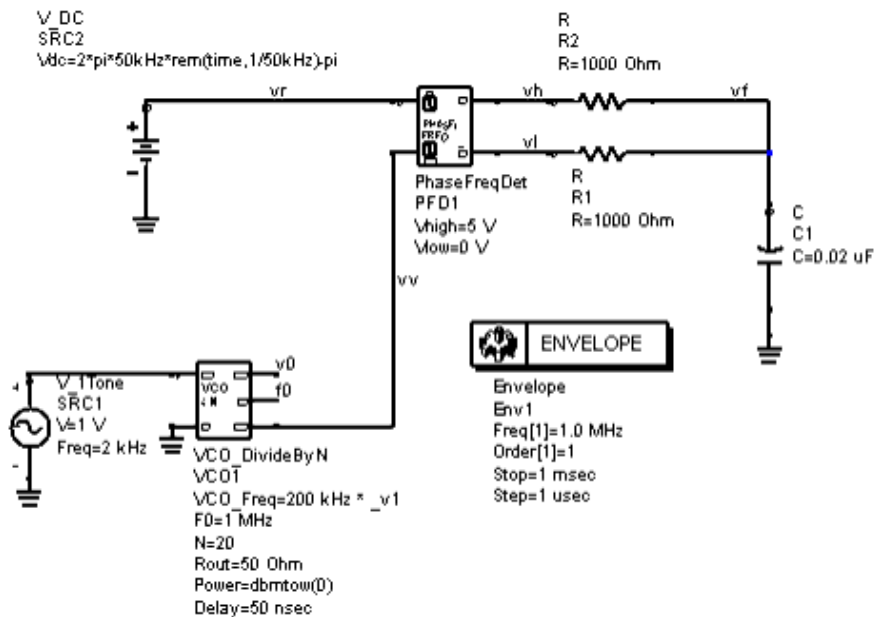


**PhaseFreqDet Example**

Eqn vout=real(vf[0])
Eqn vhigh=real(vh[0])
Eqn vlow=real(vl[0])
Eqn vref=real(vr[0])
Eqn v_vcon=real(w[0])
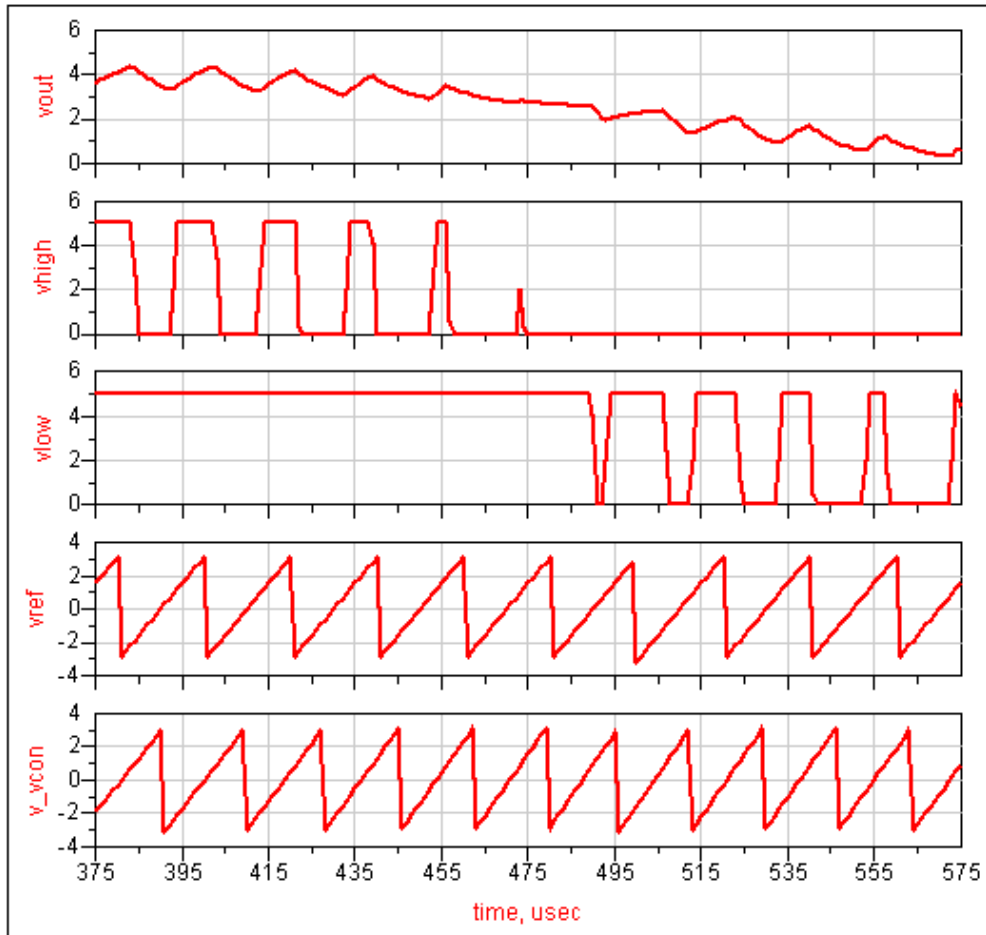


**Output Waveforms**

# PhaseFreqDetCP (Frequency Detector, Baseband with Charge Pump)

## Symbol



## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Ihigh | High-level charge pump current | mA | 1 |
| Ilow | Low-level charge pump current | mA | 1 |
| DeadTime | Dead-zone pulse width | psec | 0 |
| Jitter | Input time jitter | psec | 0 |

## Notes/Equations

1. This baseband phase-frequency demodulator with charge pump is used in transient or circuit envelope simulation.
2. Charge pump currents can be constants (in which case they are ideal current sources) or functions of the state variable _v2 to allow for a non-ideal current whose peak value varies as a function of the output voltages. In either case, the off-state current is always 0.0. Normally positive values would be used for both currents, as Ihigh specifies a source current and Ilow specifies the sink current.
3. DeadTime specifies the period of time centered around the 0 phase output during which no output is generated from either the high or the low charge pump. Outside of this period, the output returns to the ideal pulse widths and amplitudes determined by the trigger crossings of the inputs and the charge pump currents.
4. The Jitter value defines the RMS time jitter associated with the trigger crossing at each of the two inputs. The effective output time jitter will be sqrt(2) times this value. The distribution is Gaussian and the noise spectrum is assumed flat out to the reference sampling frequency.

# PhaseFreqDetTuned (Phase Frequency Detector, Tuned)

## Symbol
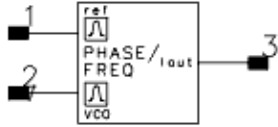


## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Sensitivity | Detector sensitivity (mA/degree) | None | 0.1 |
| MaxAngle | Maximum unwrapped phase angle (+/− MaxAngle) | deg | 360 |
| Vlimit | Maximum output voltage compliance (+/− Vlimit) | V | 20 |
| Fnom | Nominal input frequency for VCO and REF inputs | kHz | 100 |

## Notes/Equations

1. This tuned phase-frequency demodulator is used with circuit envelope simulation that models the ideal behavior of the phase frequency detectors used in phase-locked loops. (This model does not work with transient simulation.)
   On the REF and the VCO inputs, it selects the input carrier closest to the specified Fnom frequency. For proper operation, this cannot be the baseband (dc) envelope. It then generates a baseband output signal equal to the phase difference between the VCO and REF inputs, with an offset of $2 \times \pi \times \Delta f \times time$ being generated if there is a frequency difference $\Delta f$ between the two inputs.
   As an example consider the sources
   $V_1(t) = \exp(j \times P_1(t))$ at $f_0$ and
   $V_2(t) = \exp(j \times P_2(t))$ also at $f_0$,
   as input to a PhaseFreqDetTuned component with the output terminated in a high impedance.

   > **ℹ NOTE** Pn(t) is a function that is defined by phase or a phase descriptor.

   Envelope simulation at $f_0$ yields a DC output with a $P_1(t) - P_2(t)$ time dependence. Changing the frequency of the second source from $f_0$ to $f_0 + \Delta f$ for the same $V_2(t)$ is equivalent with keeping the frequency at $f_0$ but multiplying $V_2(t)$ by $\exp(j \times 2 \times \pi \times \Delta f \times t)$ and results in the output signal $P_1(t) - P_2(t) - (2 \times \pi \times \Delta f \times t)$.
   If there is no analysis harmonic frequency close enough to the Fnom frequency, then a warning is generated. The two input impedances are infinite. The output is a baseband-only infinite impedance current source with scaling defined by the Sensitivity parameter. If the output voltage exceeds Vlimit or is lower than −Vlimit, then a voltage limiter with an impedance of 10 Ohms is switched in.
   The MaxAngle parameter can be used to model special phase-frequency detectors

that have greater than the traditional ±2π of input phase range.

2. This model does not model any of the reference or VCO frequency components on the output. The output is a baseband-only signal, and the voltage limiting applies to the baseband component only. (See the baseband phase-frequency detector for a model that can overcome this limitation.) With this tuned model, though, the timesteps can actually be larger than the reference period, and it is only dependent on the PLL bandwidth and not the reference frequency; this enables faster performance for many types of simulation measurements.

Another limitation is that only the phase of the selected input envelope is included. The effect of additional input spurs and harmonics are not included if they are in different carrier frequency envelopes.

3. Phase-Frequency Detector Example and Simulation Results show a simple schematic and example of this phase-frequency detector's output. The reference input is a fixed frequency. The VCO input has sinusoidal frequency modulation.
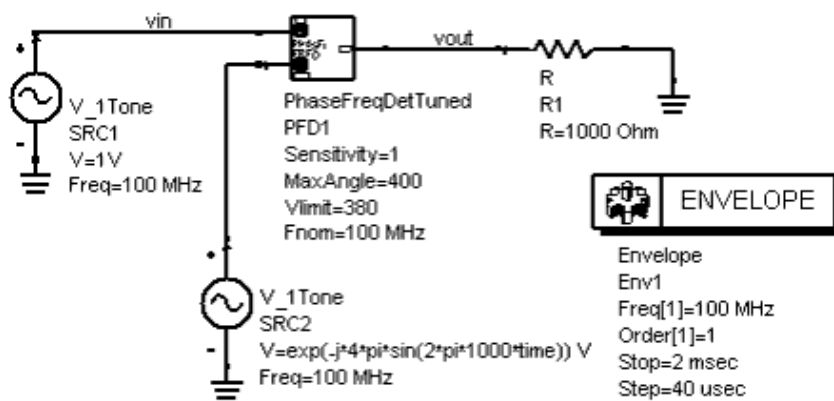


**Phase-Frequency Detector Example**
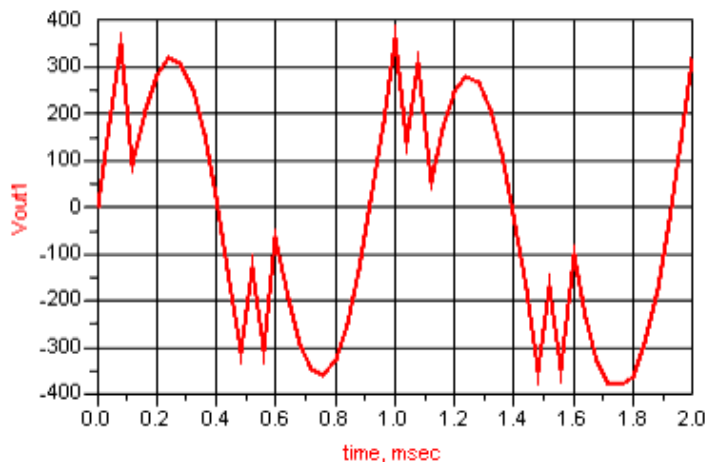


**Simulation Results**

# PhaseNoiseMod (Phase Noise Modulator)

## Symbol



## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Fnom | Nominal input frequency | GHz | 1 |
| Rout | Output resistance | Ohm | 50 |
| Fcorner | Corner frequency for 1/f noise performance | MHz | 1 |
| NF | Broadband noise figure | dB | 3 |
| QL | Loaded Q of resonator | None | 500 |

## Notes/Equations

1. This device uses Leeson's equation to model oscillator phase noise, then modulates the input carrier with this phase noise.
   The input can be from any signal source, including the VCO models. This model behaves as a tuned modulator by selecting and modulating just the carrier defined by the Fnom parameter. If there are no analysis frequencies close enough to this value, a warning is generated and no signal is output. The Leeson's equation models the oscillator phase using the equation

   $$S_{\Delta\phi} = \frac{(F)kTB}{P_{sig_{AV}}} \times \left(1 + \left(\frac{F_{nom}}{2Q_L Freq}\right)^2\right) \times \left(1 + \frac{F_{corner}}{Freq}\right)$$

   where
   Fnom is used as an approximation to carrier frequency
   F is the noise factor

   $$P_{sig_{AV}}$$

   is the input signal power
   T is the absolute temperature
   B is the analysis bandwidth
   k is the Boltzmann constant
   $Q_L$ is the loaded Q value of the oscillator's resonator

   Fcorner is the frequency at which the low frequency 1/f noise is equal to the broadband noise.
   This model is usable in frequency domain and circuit envelope time domain noise analyses. To avoid the divide-by-zero problems as the analysis offset frequency approaches 0, both the $1/f^2$ and 1/f terms are rolled off at frequencies below 1 Hz. In the time-domain mode, the 1/f frequency response is implemented by doing a

convolution simulation. The duration of this impulse response is set to 2000 timesteps. This effectively rolls off this 1/f response at a frequency determined by the analysis tstep parameter.

2. This circuit phase modulates one large signal frequency with noise to produce pure phase noise without amplitude noise. Because of the nature of harmonic balance, the noise exists at sidebands above and below this one large signal tone, but around no other large signal frequencies. Thus, if phase noise is specified at an offset frequency large enough that it could appear around another large signal frequency, the noise will not appear if viewed around that other large signal frequency. This can be a limitation when attempting to combine two sources that are close together in frequency. In that case, the P_1Tone with phase noise should be used instead for harmonic balance analysis.

# VCO (Voltage Controlled Oscillator)

## Symbol



## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Kv | Frequency tuning sensitivity (Hz/Volt) | kHz | 1 |
| Freq | Fundamental frequency | GHz | 1 |
| P | Power into Rout Load at Fundamental Frequency, complex value | None | $-j \times$ dbmtow(0) |
| Rout | Output resistance | Ohm | 50 |
| Delay | Transit time delay added to input tuning voltage | None | timestep |
| Harmonics | Ratio of harmonic voltage to fundamental voltage, complex value | None | list(-0.01, 0.002) |

## Notes/Equations

1. This is a simple VCO model that outputs a signal whose frequency is controlled in a linear manner by the input tuning voltage. The center frequency is defined by the Freq parameter, which references one of the analysis frequencies. A Freq value of 0 references dc and so defines the VCO as a baseband source; the output frequency is the baseband input voltage times the Kv parameter value. Only the baseband portion of the input tuning voltage is used to determine the VCO frequency offset.

2. The phase of the VCO output is clamped when time equals 0, so this model only functions as a VCO in the time-domain analysis modes, including circuit envelope and transient simulation. In a transient simulation, the VCO *Freq* parameter can be 0, or set to a value that is the VCO's nominal frequency.

3. The output resistance at all frequencies is set by Rout, which is internally limited to a minimum value of 0.1 Ohm. The VCO's fundamental output power into an Rout load is defined by the P parameter. The relative level, in linear units, of the second and third harmonics are defined by the Harmonics parameter; this parameter can be complex to allow definition of the relative phase of these harmonics. The initial phase for the fundamental frequency output is set to $-90$ degrees so that a sine waveform is created in baseband mode.

4. A Delay parameter value can also be specified for this VCO model. This puts an additional transit delay between the input tuning voltage and the actual change in the output frequency. A delay of at least one time-step does sometimes result in faster simulation speeds, and can be used to model the time delay inherent in any real VCO.

   Care should be taken when using the VCO in baseband mode (Freq = 0). If the input voltage is allowed to go negative, the model will generate a negative frequency, and the waveform shape due to the harmonic content will be reversed.

The VCO_DivideByN model also allows the definition of a nonlinear frequency tuning characteristic.

5. The following figure shows the use of the VCO in an envelope simulation. The VCO is defined with significant second and third harmonic levels. Simulation results are shown in the next figure.

ENVELOPE

Envelope
Env1
Freq[1]=1.0 GHz
Order[1]=0
Stop=500 usec
Step=1 usec

mod    vco    vout

VtPulse
SRC1
Vlow=0 V
Vhigh=3 V
Delay=0 nsec
Edge=linear
Rise=400 usec
Fall=100 usec
Width=0 nsec
Period=

VCO
VCO1
Kv=20 kHz
Freq=0 GHz
P=dbmtow(0)
Rout=50 Ohm
Delay=timestep
Harmonics=list(0.2*j, -0.2*j)

R
R1
R=50 Ohm
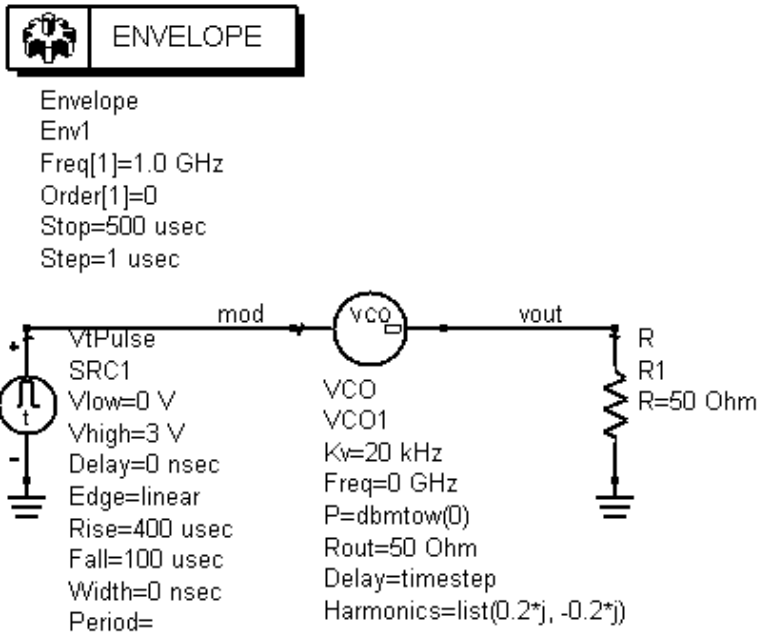
**VCO in a circuit envelope simulation**

Eqn mod_out=real(mod[0])
Eqn vout_out=real(vout[0])
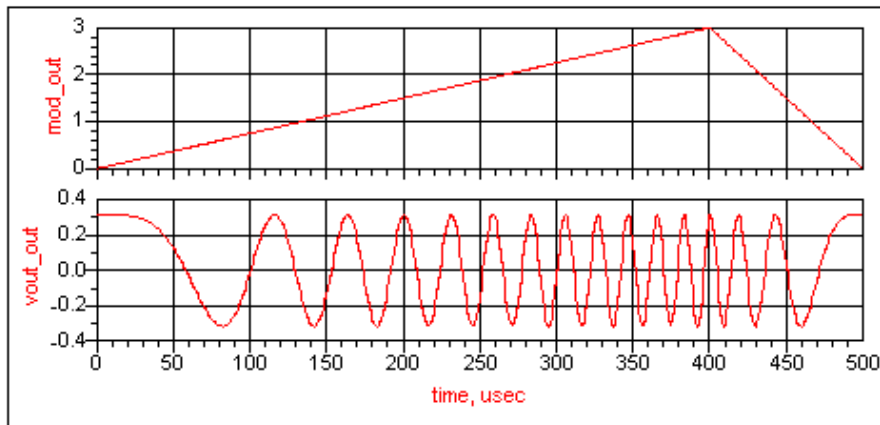


**Simulation Results**

# VCO_DivideByN (VCO Divide By N)

## Symbol
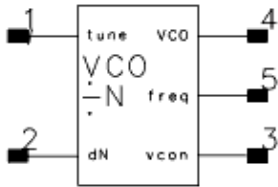


## Parameters

| Name | Description | Units | Default |
|---|---|---|---|
| VCO_Freq | Frequency deviation from F0 (function of _v1) | None | 10 MHz × _v1 |
| F0 | VCO center frequency | GHz | 1 |
| N | Nominal divide number (with dN=0) | None | 10 |
| Rout | Output resistance of VCO | Ohm | 50 |
| Power | Output power into Rout load | dBM | 0 |
| Delay | Transit time delay added to input tuning voltage | nsec | 50 |

## Pin Connections

tune connects to a tuning voltage.

dN connects to ground (dN=0) or a voltage source such as V_DC.

dN takes the value of nodal voltage and N+dN becomes the divide ratio.

vcon is the divided-by-(N+dN) output.

freq outputs the undivided frequency values from pin VCO. This pin can be left open.

VCO is the undivided VCO output. This pin can be left open if an undivided VCO is not used.

## Notes/Equations

1. This VCO model allows for the definition of an arbitrary, nonlinear frequency tuning characteristic. In addition, it incorporates a behavioral, divide by N model. Incorporating the divider into the same model permits its use in phase-lock loop simulations where the envelope bandwidth, as determined by the analysis time-step, does not have to include the entire tuning range of the VCO, but only the frequency range of the divided output. In these cases, the phase and frequency information of the VCO's main output may be aliased because of the large time-step. But, if just the divided output is being used, the loop simulations will still be valid and can simulate faster due to the large time-step.

   As the time-step is decreased, or the range of the VCO is reduced, such that it remains within the envelope bandwidth, then both the main VCO and the divided VCO output are valid. As with the standard VCO model, the phase of this model's outputs

are also clamped when time=0 so this model only functions as a VCO in the time-domain analysis modes, including circuit envelope and transient simulations.

2. The frequency of the VCO is determined by the F0 value plus the VCO_Freq value. The VCO_Freq value may be an arbitrary expression using _v1, which is a pre-defined variable representing the input tuning voltage. The frequency is determined inside the model by determining the present value of _v1 (the input voltage), evaluating the VCO_Freq expression and adding this to the F0 value. This frequency is output, as an ideal voltage source scaled to 1V per GHz, on the freq output pin.

3. In circuit envelope simulation, the carrier frequency envelope associated with the main VCO frequency is determined by the F0 value. If there is no analysis harmonic frequency close enough to F0, then a warning is generated and the main output is zero. The carrier frequency envelope associated with the divided VCO frequency is determined by F0/N. Again, if no analysis harmonic frequency is close enough, then a warning is generated and this output is also set to zero.

4. The divide number is determined by adding the N parameter and the dN baseband input voltage. The divide number can change during the simulation. By properly driving the dN input, fractional frequency division can be simulated. To simulate all the dynamics of a fractional divider, the simulation time-step must be small enough to properly digitize the varying divide- or pulse-swallowing rate. Alternatively, either the N value or the dN input can be set to fractional values to obtain a steady-state, fractional division that would not include the switching dynamics and spurs. In circuit envelope simulation, the divided VCO frequency must remain within its initial envelope bandwidth for all combinations of VCO frequencies and divide numbers. It will not automatically jump from one envelope carrier frequency to another. In transient simulation, because everything is treated as baseband signals, the only constraint is that the time-step must be small enough to cover the maximum frequency.

5. Both the main VCO output and the divided output have an output resistance set by the Rout parameter. The main VCO output will deliver the specified Power into a Rout load. The divided output will also deliver this amount of power i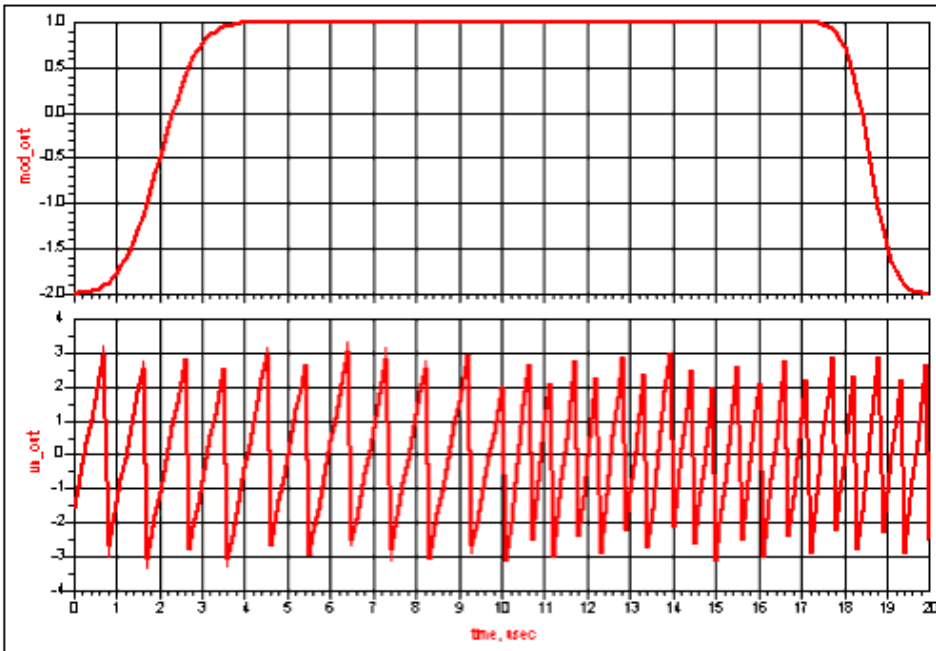f it is not a baseband output. If it is a baseband output, then the divided output is a sawtooth waveform, whose open circuit voltage represents the instantaneous phase, in radians, of the divided signal. In transient simulation, then, this divided output is always a sawtooth. In circuit envelope, it is a sawtooth if F0/N is within the baseband envelope (it is less than 0.5/timestep). If F0/N is closer to one of the analysis carrier frequencies, then the output is a complex sinusoid with the same amplitude as the main VCO output.

6. A Delay parameter value can also be specified for this VCO model. This puts an additional transit delay between the input tuning voltage and the actual change in the output frequency. A delay of at least one timestep does sometimes result in slightly faster simulation speeds, and can be used to model the time delay inherent in any real VCO.

7. Care should be taken when using the VCO in baseband mode. If F0+VCO_Freq value goes negative, the model will generate a *negative* frequency, which may give unexpected results. If this is a problem, the VCO_Freq expression could include a limiting operator to prevent this.

8. Example with Linear Tuning Characteristics shows an example application; simulation results are shown in Simulation Results. The tuning characteristic is linear in this case and is simply 1 MHz per volt. The nominally divided output frequency is 100MHz/55=1.81818 MHz, so this output can be a baseband output, given the 0.1 μ timestep. The divide number changes from 95 to 55 halfway through the simulation. (Note that while the divided VCO output may not appear to be a regular, uniform, amplitude sawtooth waveform, it does accurately represent the divided signal's

phase.)

9. The frequency can be determined by calculating the phase slope; by using linear interpolation, the baseband phase frequency detectors can accurately determine threshold crossings.

mod | vout | fm

VtPulse
SRC2
Vlow=-2 V
Vhigh=1 V
Delay=0 nsec
Edge=erf
Rise=tstop/5
Fall=tstop/8
Width=2*tstop/3
Period=

V_DC
SRC1
Vdc=dn_div

VCO
÷N
yf
vn

VCO_DivideByN
VCO1
VCO_Freq=1MHz * _v1
F0=f1
N=55
Rout=50 Ohm
Power=dbmtow(0)
Delay=50 nsec

FM
rf

FM_DemodTuned
DEMOD1
Sensitivity=1 KHz
Fnom=f1
Rout=50 Ohm

VAR
VAR1
tstep=0.1us
tstop=200*tstep
f1=100 MHz
dn_div=if(time>tstop/2) then 0 else 40 endif

ENVELOPE

Envelope
Env1
Freq[1]=f1
Order[1]=1
Stop=tstop
Step=tstep

**Example with Linear Tuning Characteristics**

Eqn vn_out=real(vn[0])



**Simulation Results**

# Switch and Algorithmic Components

- *ClockLFSR (Linear Feedback Shift Register)* (ccsys)
- *Comparator (Comparator)* (ccsys)
- *Differentiator (Differentiator)* (ccsys)
- *DPDT Static (Double Pole Double Throw Switch, Static)* (ccsys)
- *IntegratorSML (Integrator)* (ccsys)
- *LimiterSML (Limiter)* (ccsys)
- *ParallelSerial (Parallel to Serial Shift Register)* (ccsys)
- *PathSelect2 (Coordinated set of Single Pole Double Throw Static Switches)* (ccsys)
- *QuantizerSML (Quantizer)* (ccsys)
- *ResetSwitch (Reset Switch)* (ccsys)
- *SampleHoldSML (Sample Hold)* (ccsys)
- *Sampler (Sampler)* (ccsys)
- *SerialParallel (Serial to Parallel Shift Register)* (ccsys)
- *SPDT Dynamic (Single Pole Double Throw Switch, Dynamic)* (ccsys)
- *SPDT Static (Single Pole Double Throw Switch, Static)* (ccsys)
- *SwitchV (Voltage Controlled Switch)* (ccsys)
- *SwitchV Model (Voltage Controlled Switch Model)* (ccsys)
- *VSum (Voltage Summer)* (ccsys)

# ClockLFSR (Linear Feedback Shift Register)

## Symbol

## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Vlow | Lower threshold voltage | V | 0.25 |
| Vhigh | Upper threshold voltage | V | 0.75 |
| Taps | Bits used to generate feedback | None | bin("10000100") |
| Seed | Initial value loaded into shift register | None | bin("10101010") |
| Rout | Output resistance | Ohm | 1 |

## Notes/Equations

1. ClockLFSR can be used to generate PN sequences with user-defined recurrence relations. The input is a clock signal; with each positive clock edge, the next output bit is calculated. A clock edge occurs any time the baseband input signal rises through 0.5V.
2. This model works in transient and circuit envelope simulation.
3. With each positive clock edge, data is shifted to the right in the shift register. The length of the shift register is determined by the most significant one-bit in the Taps value.
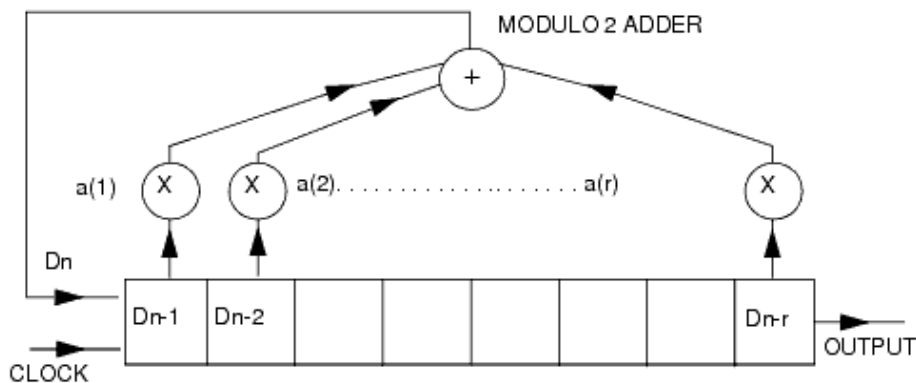


## LFSR Model

The numbers a(1), a(2), … , a(r) are the binary feedback coefficients and are specified by

the Taps parameter. This value may be specified as a decimal number, or as a binary pattern if the bin( ) function is used. The initial contents of the shift are specified by the value of the Seed parameter. The following equations describe the operation of this component:

At each positive clock edge n (n ≥ 1),

for n ≥ 1:

$$D(n) = \left[ \sum_{k=1}^{r} a(k)D(n-k) \right] mod\ 2$$

where

D(0) = Seed$_2$ (0)

D(−1) = Seed$_2$ (1)

.

.

.

D(1−r) = Seed$_2$ (r−1)

$$\sum_{k>o} Seed_2(k)2^k$$

and Seed =

where Seed$_2$(k) Œ{0,1} for 0 ≤ k < r.

Example: Let Seed = 2, and Taps = 7

then

Seed$_2$ (0) = 0

Seed$_2$ (1) = 1

.

.

.

Seed$_2$ (6) = 0

thus,

(0) = Seed2 (0) = 0
D(-1) = Seed2 (1) = 1
D(-2) = Seed2 (2) = 0

.

.

.

D(-6) = Seed$_{22}$ (6) = 0

Seed2 (2) = 0

Linear Feedback Shift Register ClockLFSR

The binary feedback coefficients are specified by Taps. For example, the recurrence relation

D(n) = (D(n-7) + D(n-3) + D(n-2) + D(n-1))mod 2

is specified by:

Taps = bin("1000111")

or

Taps = 71

The following table provides a list of feedback coefficients for linear feedback shift registers showing one or more alternate feedback connections for a given number of

stages.

**Feedback Connections for Linear m-Sequences\***

| No. of Stages | Code Length | Maximal Taps |
|---|---|---|
| 2 | 3 | [2, 1] |
| 3 | 7 | [3, 1] |
| 4 | 15 | [4, 1] |
| 5 | 31 | [5, 2] [5, 4, 3, 2] [5, 4, 2, 1] |
| 6 | 63 | [6, 1] [6, 5, 2, 1,] [6, 5, 3, 2,] |
| 7 | 127 | [7, 1] [7, 3] [7, 3, 2, 1,] [7, 4, 3, 2,] [7, 6, 4, 2] [7, 6, 3, 1] [7, 6, 5, 2][7, 6, 5, 4, 2, 1] [7, 5, 4, 3, 2, 1] |
| 8 | 255 | [8, 4, 3, 2] [8, 6, 5, 3] [8, 6, 5, 2] [8, 5, 3, 1] [8, 6, 5, 1] [8, 7, 6, 1][8, 7, 6, 5, 2, 1] [8, 6, 4, 3, 2, 1] |
| 9 | 511 | [9, 4] [9, 6, 4, 3] [9, 8, 5, 4] [9, 8, 4, 1] [9, 5, 3, 2] [9, 8, 6, 5] [9, 8, 7, 2][9, 6, 5, 4, 2] [9, 7, 6, 4, 3, 1] [9, 8, 7, 6, 5, 3] |
| 10 | 1023 | [10, 3] [10, 8, 3, 2] [10, 4, 3, 1] [10, 8, 5, 1][10, 8, 5, 4] [10, 9, 4, 1][10, 8, 4, 3] [10, 5, 3, 2] [10, 5, 2, 1] [10, 9, 4, 2] |
| 11 | 2047 | [11, 1] [11, 8, 5, 2] [11, 7, 3, 2] [11, 5, 3, 5] [11, 10, 3, 2] [11, 6, 5, 1][11, 5, 3, 1] [11, 9, 4, 1] [11, 8, 6, 2] [11, 9, 8, 3] |
| 12 | 4095 | [12, 6, 4, 1] [12, 9, 3, 2] [12, 11, 10, 5, 2, 1] [12, 11, 6, 4, 2, 1][12, 11, 9, 7, 6, 5] [12, 11, 9, 5, 3, 1] [12, 11, 9, 8, 7, 4] [12, 11, 9, 7, 6, 5] [12, 9, 8, 3, 2, 1] [12, 10, 9, 8, 6, 2] |
| 13 | 8191 | [13, 4, 3, 1] [13, 10, 9, 7, 5, 4] [13, 11, 8, 7, 4, 1] [13, 12, 8, 7, 6, 5][13, 9, 8, 7, 5, 1] [13, 12, 6, 5, 4, 3] [13, 12, 11, 9, 5, 3] [13, 12, 11, 5, 2, 1] [13, 12, 9, 8, 4, 2] [13, 8, 7, 4, 3, 2] |
| 14 | 16,383 | [14, 12, 2, 1] [14, 13, 4, 2] [14, 13, 11, 9] [14, 10, 6, 1] [14, 11, 6, 1][14, 12, 11, 1] [14, 6, 4, 2] [14, 11, 9, 6, 5, 2] [14, 13, 6, 5, 3, 1] [14, 13, 12, 8, 4, 1] [14, 8, 7, 6, 4, 2] [14, 10, 6, 5, 4, 1] [14, 13, 12, 7, 6, 3] [14, 13, 11, 10, 8, 3] |
| 15 | 32,767 | [15, 13, 10, 9] [15, 13, 10, 1] [15, 14, 9, 2] [15, 1] [15, 9, 4, 1] [15, 12, 3, 1] [15, 10, 5, 4] [15, 10, 5, 4, 3, 2] [15, 11, 7, 6, 2, 1] [15, 7, 6, 3, 2, 1] [15, 10, 9, 8, 5, 3] [15, 12, 5, 4, 3, 2] [15, 10, 9, 7, 5, 3] [15, 13, 12, 10] [15, 13, 10, 2] [15, 12, 9, 1] [15, 14, 12, 2] [15, 13, 9, 6] [15, 7, 4, 1] [15, 4] [15, 13, 7, 4] |
| 16 | 65,535 | [16, 12, 3, 1] [16, 12, 9, 6] [16, 9, 4, 3] [16, 12, 7, 2] [16, 10, 7, 6][16, 15, 7, 2] [16, 9, 5, 2] [16, 13, 9, 6] [16, 15, 4, 2] [16, 15, 9, 4] |
| 17 | 131,071 | [17, 3] [17, 3, 2] [17, 7, 4, 3] [17, 16, 3, 1] [17, 12, 6, 3, 2, 1][17, 8, 7, 6, 4, 3] [17, 11, 8, 6, 4, 2] [17, 9, 8, 6, 4, 1] [17, 16, 14, 10, 3, 2] [17, 12, 11, 8, 5, 2] |
| 18 | 262, 143 | [18, 7] [18, 10, 7, 5] [18, 13, 11, 9, 8, 7, 6, 3] [18, 17, 16, 15, 10, 9, 8, 7] [18, 15, 12, 11, 9, 8, 7, 6] |
| 19 | 524,287 | [19, 5, 2, 1] [19, 13, 8, 5, 4, 3] [19, 12, 10, 9, 7, 3] [19, 17, 15, 14, 13, 12, 6, 1] [19, 17, 15, 14, 13, 9, 8, 4, 2, 1] [19, 16, 13, 11, 19, 9, 4, 1] [19, 9, 8, 7, 6, 3] [19, 16, 15, 13, 12, 9, 5, 4, 2, 1] [19, 18, 15, 14, 11, 10, 8, 5, 3, 2] [19, 18, 17, 16, 12, 7, 6, 5, 3, 1] |
| 20 | 1, 048,575 | [20, 3] [20, 9, 5, 3] [20, 19, 4, 3] [20, 11, 8, 6, 3, 2] [20, 17, 14, 10, 7, 4, 3, 2] |
| 21 | 2,097,151 | [21, 2] [21, 14, 7, 2] [21, 13, 5, 2] [21, 14, 7, 6, 3, 2] [21, 8, 7, 4, 3, 2][21, 10, 6, 4, 3, 2] [21, 15, 10, 9, 5, 4, 3, 2] [21, 14, 12, 7, 6, 4, 3, |

| | | 2][21, 20, 19, 18, 5, 4, 3, 2] |
|---|---|---|
| 22 | 4,194,303 | [22,1] [22, 9, 5, 1] [22, 20, 18, 16,6, 4, 2, 1] [22, 19, 16, 13, 10, 7, 4, 1][22, 17, 9, 7, 2, 1] [22, 17, 13, 12, 8, 7, 2, 1] [22, 14, 13, 12, 7, 3, 2, 1] |
| 23 | 8,388,607 | [23, 5] [23, 17, 11, 5] [23, 5, 4, 1] [23, 12, 5, 4] [23, 21, 7, 5][23, 16, 13, 6, 5, 3] [23, 11, 10, 7, 6, 5] [23, 15, 10, 9, 7, 5, 4, 3][23, 17, 11, 9, 8, 5, 4, 1] [23, 18, 16, 13, 11, 8, 5, 2] |
| 24 | 16,777,215 | [24, 7, 2] [24, 4, 3, 1] [24, 22, 20, 18, 16, 14, 11, 9, 8, 7, 5, 4][24, 21, 19, 18, 17, 16, 15, 14, 13, 10, 9, 5, 4, 1] |
| 25 | 33,554, 431 | [25, 3] [25, 3, 2, 1] [25, 20, 5, 3] [25, 12, 5, 4] [25, 17, 10, 3, 2, 1][25, 23, 21, 19, 9, 7, 5, 3] [25, 18, 12, 11, 6, 5, 4] [25, 20, 16, 11, 5, 3, 2, 1] [25, 12, 11, 8, 7, 6, 4, 3] |
| 26 | 67,108,863 | [26, 6, 2, 1] [26, 22, 21, 16, 12, 11, 10, 8, 5, 4, 3, 1] |
| 27 | 134,217,727 | [27, 5, 2, 1] [27, 18, 11, 10, 9, 5, 4, 3] |
| 28 | 268,435,455 | [28, 3] [28, 13, 11, 9, 5, 3] [28, 22, 11, 10, 4, 3] [28, 24, 20, 16, 12, 8, 4, 3, 2, 1] |
| 29 | 536,870,911 | [29, 2] [29, 20, 11, 2] [29, 13, 7, 2] [29, 21, 5, 2] [29, 26, 5, 2] [29, 19, 16, 6, 3, 2] [29, 18, 14, 6, 3, 2] |
| 30 | 1,073,741,823 | [30, 23, 2, 1] [30, 6, 4, 1] [30, 24, 20, 16, 14, 13, 11, 7, 2, 1] |
| 31 | 2,147,483,647 | [31, 29, 21, 17] [31, 28, 19, 15] [31, 3] [31, 3, 2, 1] [31, 13, 8, 3] [31, 21, 12, 3, 2, 1] [31, 20, 18, 7, 5, 3] [31, 30, 29, 25] [31, 28, 24, 10][31, 20, 15, 5, 4, 3] [31, 16, 8, 4, 3, 2] |
| 32 | 4,294,967,295 | [32, 22, 2, 1] [32, 7, 5, 3, 2, 1] [32, 28, 19, 18, 16, 14, 11, 10, 9, 6, 5, 1] |
| 33 | 8,589,934,591 | [33, 13] [33, 22, 13, 11] [33, 26, 14, 10] [33, 6, 4, 1] [33, 22, 16, 13, 11, 8] |
| 61 | 2,305,843,009, 213, 693, 951 | [61, 5, 2, 1] |
| 89 | 618,970,019,642,690, 137,449,562,112 | [89, 6, 5, 3] |

# Comparator (Comparator)

## Symbol
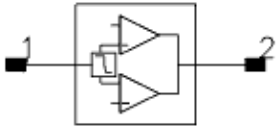
## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Vlow | Lower threshold voltage | V | 0.25 |
| Vhigh | Upper threshold voltage | V | 0.75 |

## Notes/Equations

1. Comparator outputs a 1V signal whenever the baseband portion of the input signal is between the two threshold voltages. Input impedance is infinite; output impedance is fixed at 0.1 Ohm; and, there is a fixed delay of one timestep.
2. If the baseband portion of Vin is greater than Vlow and less than Vhigh, then the output voltage at the next time sample is 1.0V; otherwise, the output is 0.0V.
3. This model works in transient and circuit envelope simulation.

# Differentiator (Differentiator)

**Symbol**
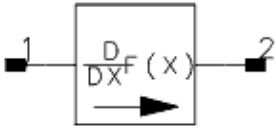


**Parameters**

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Gain | Differentiator slope | None | 1 |
| Rref | Reference resistance for both ports | Ohm | 50 |

**Notes/Equations**

1. The output voltage is equal to the derivative (with respect to time) of the input voltage.
2. The Gain parameter is entered as a linear quantity, not in dB.
3. Differentiator works in both transient and envelope simulations.
4. The input resistance of the integrator is Rref. The output is a voltage source Vout in series with output resistance Rref.

# DPDT_Static (Double Pole Double Throw Switch, Static)

## Symbol
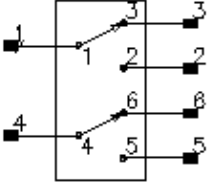


## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| State | State of switch: | None | 1 |
| F1 | First frequency breakpoint | | |
| F2 | Second frequency breakpoint | | |
| F3 | Third frequency breakpoint | | |
| Loss1 | Attenuation for frequency ≤ F1 | dB | 0 |
| Loss2 | Attenuation for F1 < frequency ≤ F2 | dB | 0 |
| Loss3 | Attenuation for F2 < frequency ≤ F3 | dB | 0 |
| Loss4 | Attenuation for frequency > F3 | dB | 0 |
| VSWR1 | Voltage standing wave ratio at both ports for frequencies ≤ F1 | None | 1 |
| VSWR2 | Voltage standing wave ratio at both ports for frequencies > F1, ≤ F2 | None | 1 |
| VSWR3 | Voltage standing wave ratio at both ports for frequencies > F2, ≤ F3 | None | 1 |
| VSWR4 | Voltage standing wave ratio at both ports for frequencies > F3 | None | 1 |
| Isolat | Isolation | dB | 100 |
| ZRef | Reference impedance for all ports | Ohm | 50 |
| Temp | Temperature | °C | None |

## Notes/Equations

1. This model works in transient and circuit envelope simulation.
2. Up to three frequency break points can be used to define four bands, with different losses and VSWR for each frequency band. Losses are entered as a positive attenuation. In other words, a 0.5 dB loss is entered as L1=0.5, not L1=−0.5. Isolation is assumed to be constant across all frequency bands.
   When using this device in a transient simulation, the switch setting must be constant.

   If loss values are specified versus frequency via a dataset and using the component DATASETVARIABLE, then a time delay through the device will be introduced when using transient analysis.
   Transient Analysis Time Delay = (1/(2 × MAX_FREQUENCY)) × POINTS

where
POINTS = (FILTER_ORDER/2) − 1 for FILTER_ORDER even
= (FILTER_ORDER + 1)/2 − 1 for FILTER_ORDER odd

# IntegratorSML (Integrator)

**Symbol**
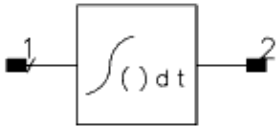


**Parameters**

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| GainAC | Integrator gain | None | 1 |
| GainDC | Gain of DC constant | None | 1.e6 |
| Rref | Reference resistance for both ports | Ohm | 50 |

**Notes/Equations**

1. The output voltage is equal to the integral (with respect to time) of the input voltage. This device has been modeled after a physically realizable integrator. The dc gain is not infinite and is specified by GainDC.
2. GainAC is the ac gain of the integrator (typically set to 1.0); GainDC is the dc gain of the integrator. Both of these are a linear quantity, not in dB.
3. An ideal integrator has infinite gain at dc, which is not physically realizable. These parameters describe the operation of the integrator that has been modeled after a traditional high gain differential amplifier with capacitive feedback and a series input resistor. GainDC is the open loop dc gain of the amplifier in this model. Gain is equal to 1/RC where R is the input series resistor value in Ohm and C is the feedback capacitor value in farads.
4. The integrator transfer function follows:

$$\frac{Vout}{Vin} = \frac{GainDC}{1 + j\omega * GainDC/GainAC}$$

   where ω is the frequency in radians/sec and Vout is the output voltage when the component has an external load of Rload=Rref. When Rload is different from Rref, the actual output voltage is

$$2Vout * Rload/(Rload + Rref)$$

5. IntegratorSML works in both transient and envelope simulations.
6. The input resistance of the integrator is Rref. The output is a voltage source Vout in series with output resistance Rref.

# LimiterSML (Limiter)

## Symbol
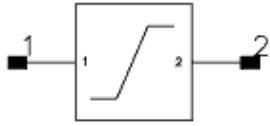


## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Gain | Linear gain (real only) | None | 1.0 |
| Vmax | Maximum output voltage | V | 5 |
| Vmin | Minimum output voltage | V | −5 |
| Rout | Output resistance | Ohm | 50 |
| Fnom | Nominal input frequency | Hz | 0 |

## Range of Usage

Fnom $\geq$ 0

## Notes/Equations

1. The Gain parameter is a real value only (not in dB or complex).
2. This model can be used in transient, harmonic balance and circuit envelope simulations.
3. When used in transient simulations, Fnom must be set to zero. Gain, Vmin and Vmax determine the output voltage time series. The output voltage time series is calculated as Gain times the input voltage time series, but is hard limited to Vmin and Vmax in case the value drops below Vmin or rises above Vmax. Hard limiting is not allowed to reduce the magnitude of the output waveform amplitude by more than a factor of $10^{-4}$ at any point in time. For example, a sine wave with a $10^6$ V amplitude as input to a LimiterSML component with Gain=10, Vmax=1, Vmin=-1, will scale to an output waveform amplitude of $10^3$ V, not 1 V. The value $10^3$ comes from $10^6$ x Gain x $10^{-4}$, where Gain=10 in this example.
4. When used in harmonic balance simulations, Fnom must be set to a non-zero fundamental or intermodulation frequency. Fnom, Gain and abs(Vmax) determines the output voltage spectrum (Vmin is not used). The output voltage spectrum is calculated as Gain times the input voltage spectrum but is scaled by min(abs(Vmax)/abs(V(Fnom)),1) at all frequencies, where V(Fnom)=Gain x Vin(Fnom). This gain limits the entire frequency spectrum by abs(Vmax)/abs(V(Fnom)) in the case where abs(V(Fnom)) exceeds abs(Vmax). If there is no spectral component at Fnom, no gain limiting is performed. No minimum

gain of $10^{-4}$ applies in this case.

5. When used in circuit envelope simulations, Fnom can be a zero or non-zero fundamental or intermodulation frequency.
    - If Fnom is zero, the mode of operation is similar to that for transient analysis. If there is a baseband signal, it is hard limited as dictated by Vmin and Vmax. Hard limiting is not allowed to drop the magnitude of the output voltage by more than a factor of $10^{-4}$ at any time (refer to note 3). All other spectral components are scaled accordingly. If there is not a baseband signal, nothing happens.
    - If Fnom is a non-zero fundamental or intermodulation frequency, the mode of operation is similar to that for harmonic balance analysis. All spectral components, including the baseband signal if applicable, are gain limited as dictated by min(abs(Vmax)/abs(V(Fnom)),1). For harmonic balance analysis, the value of Vmin does not matter in this case.

6. LimiterSML has more than one mode of operation. It performs clipping or limiting in a Transient or Circuit Envelope analysis, and it performs gain scaling on the spectral components in a Harmonic Balance analysis. Its mode of operation depends on the value of Fnom and the type of simulation being performed.

# ParallelSerial (Parallel to Serial Shift Register)

## Symbol
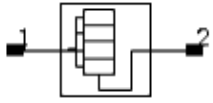


## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| OutputRate | Serial output data clock rate | kHz | 50 |
| LSB_First | Select output serial data order: | None | yes |
| Delay | Initial synchronization delay | nsec | 0.0 |
| InputBits | Number of bits in input word | None | 4 |
| IntegerIn | Scale input data as integers: yes or no | None | yes |

## Notes/Equations

1.  This Parallel To Serial Shift Register model is used to convert a sequence of input words into a serial output bit stream. The fixed serial output bit rate is specified by the OutputRate parameter. The serial data can be output with either the LSB or MSB first, depending on the state of the LSB_First parameter. The output impedance is fixed at 0.1 Ohm. A logic one generates an open circuit voltage of 1.0V. A logic zero is 0.0V.

2.  The number of bits in each input word is specified by InputBits. The input is sampled at a rate equal to OutputRate/InputBits, with an initial synchronization delay specified by Delay.

3.  The input impedance is infinite. If the IntegerIn parameter is set true, then the input is assumed to be scaled as an integer from 0.0V to $2^{OutputRate}-1$V. Otherwise, the input is assumed to be scaled from $-1.0$ to $+1.0$V, with $-1.0$V interpreted at word 0.

4.  This model works in transient and circuit envelope simulation. Only the baseband portion of circuit envelope voltage is used.

# PathSelect2 (Coordinated set of Single Pole Double Throw Static Switches)

## Symbol
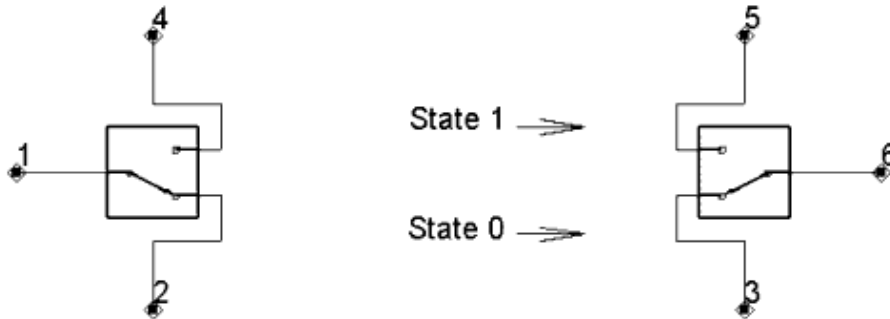


## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| State | State of switch: 0=lower path, 1=upper path | None | 0 |

## Notes/Equations

1. This model defines a pair of coordinated static single pole double throw switches that enables connectivity for one of two paths with ideal isolation between the paths.
2. This component uses internal models for SPDT_Static. The schematic design is shown here:
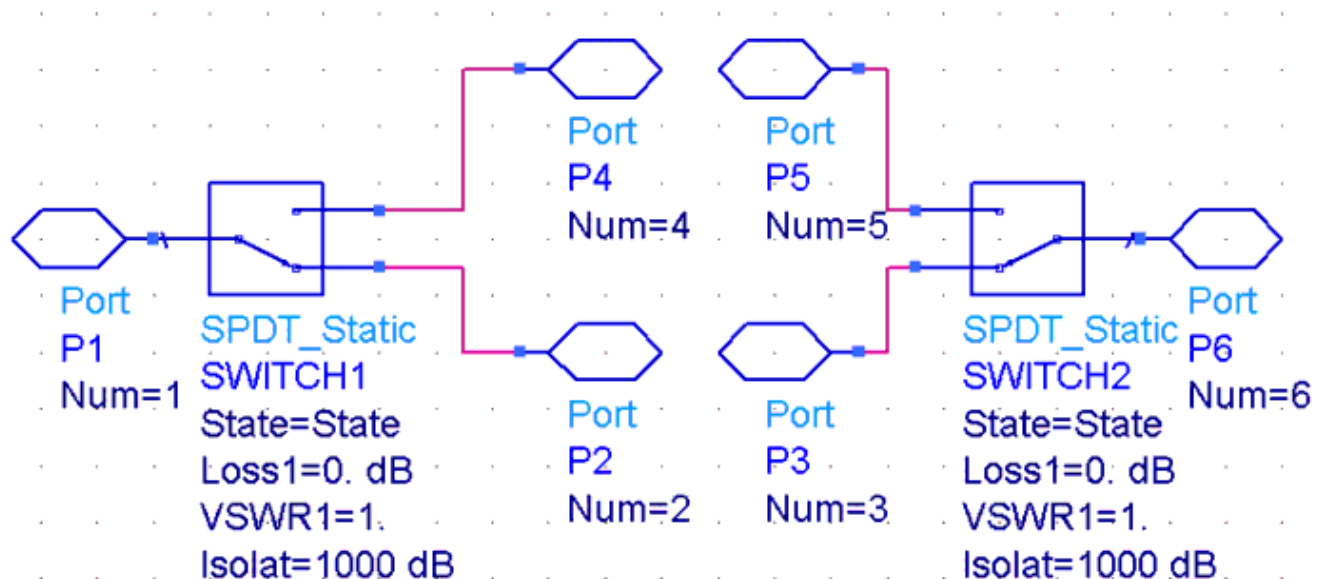


3. When State=0, connectivity occurs from port 1 to port 2 to port 3 to port 6.
4. When State=1, connectivity occurs from port 1 to port 4 to port 5 to port 6.

5. When using this device in a transient simulation, the switch setting must be constant.
6. An example demonstrating this model is located in ADS; access the design from the ADS Main window: **File** > **Open** > **Example** > **Tutorial** > *RF_Budget_Examples_wrk* ; see *Budget_Mixer* .

# QuantizerSML (Quantizer)

## Symbol
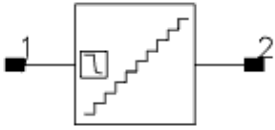


## Parameters

| Name | Description | Units | Default |
|---|---|---|---|
| Vmin | Minimum baseband input voltage | V | −1.0 |
| Vmax | Maximum baseband input voltage | V | 1.0 |
| N | Number of quantized output levels | None | 32 |
| OutState | State of output: | None | 1 |
| Rout | Output resistance | Ohm | 0.1 |

## Notes/Equations

1. This quantizer model outputs a delayed, uniformly quantized representation of the baseband portion of the input voltage. The input impedance is infinite. The delay is fixed at one timestep. The quantization state output mode is in offset binary format. The input quantization level is equal to $\Delta$ = ( *Vmax* - *Vmin* ) / ( *N* - 1). Any input less than ( *Vmin* + $\Delta$ /2) is assigned to state 0. Any input greater than ( *Vmax* - $\Delta$ /2) has a state equal to ( *N* - 1). If the OutState parameter is true, this quantization state is directly output. If the OutState parameter is equal to zero, the quantization state is scaled and offset, *Vout* = Q/$\Delta$ + *Vmin* , to match the input voltage.
2. This model works in transient and circuit envelope simulation.
3. An example of using this component along with SampleHoldSML follows.

ClockWjitter
SRC2
Low=0 V
High=1 V
Rout=1 Ohm
Delay=0 nsec
Rise=1 nsec
Fall=1 nsec
Width=25 nsec
Period=50 nsec
Jitter=0 nsec

VtStep
SRC1
Vlow=-1 V
Vhigh=1 V
Delay=0 usec
Rise=1 usec

SampleHoldSML
SAMP1
Fnom=0 Hz

QuantizerSML
QUANT1
Vmin=-1.0 V
Vmax=1.0 V
N=32
OutState=1
Rout=0.1 Ohm

TRANSIENT
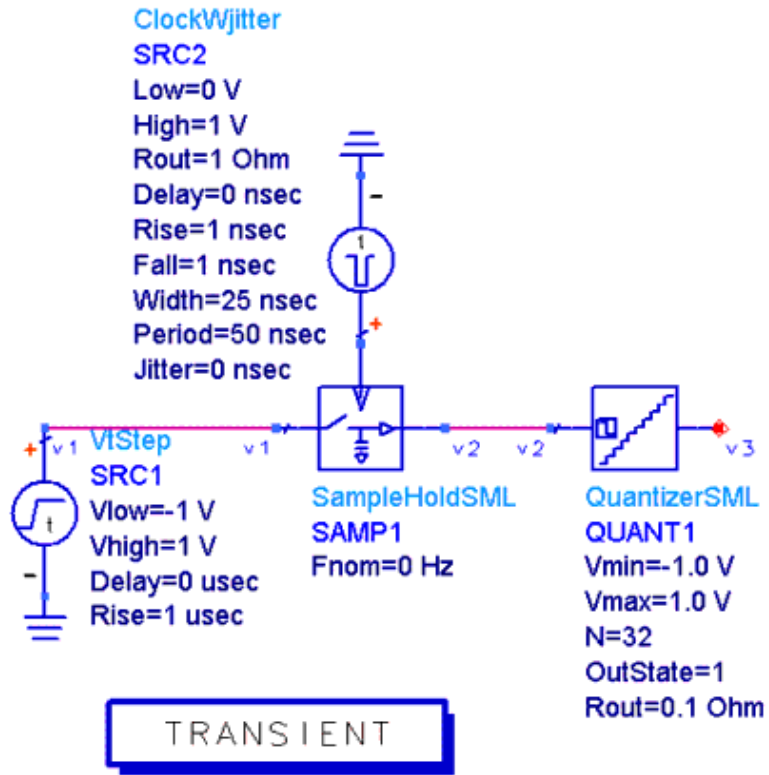
Tran
Tran1
StopTime=1 usec
MaxTimeStep=1 nsec
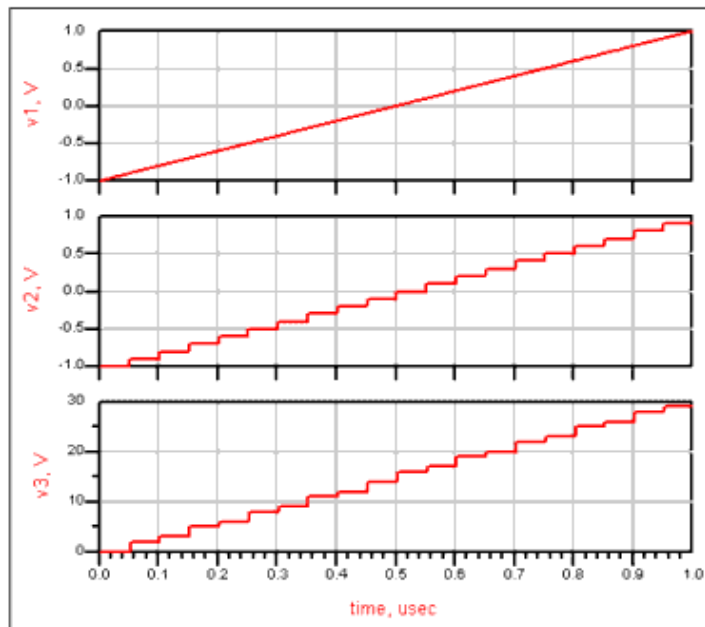
# ResetSwitch (Reset Switch)

**Symbol**
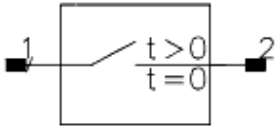


**Parameters**

None

**Notes/Equations**

1. This reset switch is an ideal switch that is closed at time=0 and open for time>0. Its main use is in time domain simulation modes (transient and circuit envelope) to allow the resetting of components such as ideal integrators. Often these blocks have infinite gain at dc which makes it difficult for the circuit simulator to converge. The ResetSwitch allows to keep them reset at time=0, so that a valid initial solution can be obtained prior to the actual time domain simulation and then the switch will automatically be opened to observe the time domain response. In steady-state analysis, such as dc or harmonic balance, time is always 0 so this switch remains closed during such analysis.

# SampleHoldSML (Sample Hold)

## Symbol
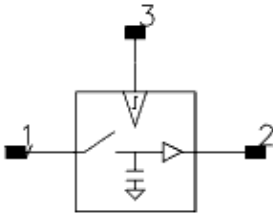


## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Fnom | Nominal input and output frequency | Hz | 0 |

## Notes/Equations

1. This model samples the input signal and holds it until the next sample event. It is a tuned model; it selects and holds just the signal at the input harmonic closest to the frequency specified by Fnom. If there is no analysis harmonic frequency close enough to the Fnom frequency, a warning is issued and the output is 0. The input impedance of both the sample clock input and the signal input is infinite. The output is a voltage source with a fixed impedance of 0.1 Ohm.

2. The sampling instant is defined to be when the sample clock input rises through the 0.5V fixed threshold. If the baseband mode of operation is specified (Fnom ~ 0 Hz), this behaves as a normal sample and hold; the baseband input voltage at the sampling instant is output and then held indefinitely at this constant value until the next sample. If an envelope carrier frequency is specified by Fnom, the signal component at this envelope harmonic frequency is selected and its complex value at the sampling instant is held at the output. This held output appears at the same envelope harmonic frequency. For example, if an envelope analysis with a fundamental of 1 GHz is performed with a timestep of 0.1nsec and an offset frequency source at 1.001 GHz is sampled, the complex value of the envelope is sampled and held constant. The output will appear to be at the 1 GHz fundamental frequency since the complex envelope is no longer changing, until the next sample event.

3. The Sampler model samples the entire spectrum and outputs just a real baseband voltage.

4. SampleHoldSML works in transient and circuit envelope simulation. In transient simulation, Fnom has no effect because all signals are baseband only.

# Sampler (Sampler)

## Symbol
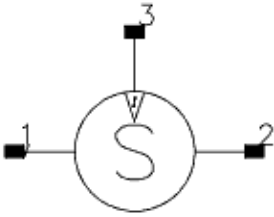


## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Ton | ON-state pulse width, switch in low impedance state | nsec | 1 |
| Ron | ON-state resistance, switch in low impedance state | Ohm | 1 |
| S11 | Input reflection coefficient | None | 0 |
| Z0 | Input port characteristic impedance | Ohm | 50 |

## Notes/Equations

1. Sampler is a linear behavioral model of a high-frequency sampler. It can also be used to model the sampling efficiency and droop of lower frequency sample-holds.
2. This model works in transient and circuit envelope simulation. In envelope simulation, the input signal is determined by transforming all the spectral input voltages at the sampling instant to determine the total, real instantaneous voltage. Prior to being sampled, the input signal is first filtered with an ideal sinc( ) filter determined by the Ton pulse width. The input match can be complex, but since it needs to represent a causal response it cannot be complex at dc.
3. The input impedance looking into the sampler is determined by Z0 and S11 with the standard formula:

$$Zin = \frac{Z0(S11+1)}{S11-1}$$

   Usually, S11=0 and Z0=50 results in the input impedance of the sampler being 50 Ohms.
4. The Clock/LO input impedance is infinite, and the sampling instant is defined only by the baseband portion of the input signal and occurs whenever the baseband signal passes through 0.5V with a positive slope.
5. Sampler has two basic modes of operation. If Ron is equal to 0, then the output is a ideal voltage source with a value equal to the last sampled value of the filtered input voltage; if Ron is not equal to 0, then the output impedance is time-varying. In the quiescent state, it is 1Tohm. However when a sample occurs, the impedance for that time point is reduced to a sampler resistance equal to Ron×timestep/Ton. The actual sampler efficiency is then determined by sampler Ron and Ton values, and the load capacitance on the sampler output.

If Ton >> Ron×CLoad, the sampler will behave with 100% efficiency. In a microwave sampler, this efficiency is typically much less than 100%. The hold time constant in this sampler mode is determined strictly by the capacitive and resistive load placed on the sampler output by the user.

The Ron sampler parameter should include all charging impedances, including the sampler switch impedance as well as the effect of any source impedance.

The output signal in Envelope mode is a baseband-only signal. No RF leakage is included in this model.

Note that due to the impulse nature of this sampler model output, the analysis integration order should be set to 1 (Backward Euler) when using this model. This is especially true when sampling rapidly changing input signals and when the sampler parameters are set for high sampling efficiency.

6.  In the circuit envelope example in Circuit Envelope Example, High-Frequency Application, the input signal is a high-frequency sawtooth waveform with 15 harmonics. Its frequency is set 10 kHz above 1 GHz. The sampler is being driven at a 500 kHz rate so the output signal should be a replicated sawtooth at 10 kHz. Simulation results for two different combinations of Ron and Ton are shown in Simulation Results. The high-efficiency mode tracks the input. The low-efficiency mode (50 Ohm, 50 psec) shows its lowpass filtering impact. Note the two different time scales: 2 nsec for the input trace and 200 µ for the output traces.

The example in Circuit Envelope Example, Low-Frequency Application shows a low-frequency application that has a significant droop due to the finite output resistance. Simulation results are shown in Simulation Results.

# High-Frequency Application



**Simulation Results**



**Circuit Envelope Example,**

# Low-Frequency Application
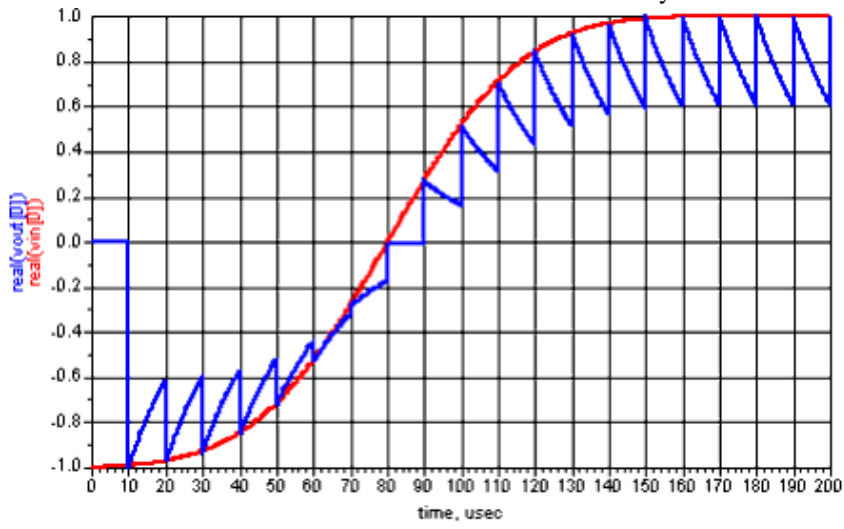
**Simulation Results**

# SerialParallel (Serial to Parallel Shift Register)

## Symbol
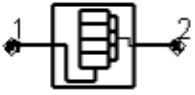


## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| InputRate | Serial input data clock rate | kHz | 50 |
| LSB_First | Order of serial data: | None | yes |
| Delay | Initial synchronization delay | nsec | 0.0 |
| OutputBits | Number of bits in output word | None | 4 |
| IntegerOut | Scale output data as integers (instead of from −1 to 1): yes or no | None | yes |

## Notes/Equations

1. This SerialParallel shift register model is used to convert a serial data stream into a multiple bit word. The fixed serial input bit rate is specified by the InputRate parameter, with an initial synchronization delay specified by the Delay parameter. The serial data can be interpreted as either LSB or MSB first, depending on the state of the LSB_First parameter. The input impedance is infinite. A voltage less than 0.5V is a logical zero.

2. The number of bits in each output word is specified by OutputBits. The output is updated at a rate equal to InputRate/OutputBits. The open circuit output voltage is an integer value from 0 to $2^{OutputBits}-1$, if the IntegerOut parameter is true. Otherwise, the output is scaled from a −1 to +1V, with all 0s corresponding to −1V and all 1s to + 1Vs. The output impedance is fixed at 0.1 Ohm.

3. This model works in transient and circuit envelope simulation. Only the baseband portion of circuit envelope voltage is used.

# SPDT_Dynamic (Single Pole Double Throw Switch, Dynamic)

## Symbol
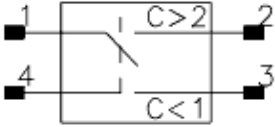


## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Ron | On-state resistance of switch | Ohm | 0 |
| Roff | Off-state resistance of switch | GOhm | 1 |

## Range of Usage
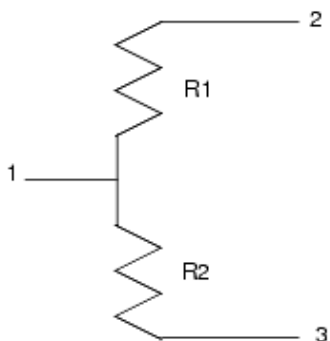


Pin 1 is the input, pins 2 and 3 are output, pin 4 is the control voltage, Vc.
If Vc > 2V, R1 = Ron, R2 = Roff
If Vc < 1v, R1 = Roff, R2 = Ron
When Vc increases from 1V to 1.5V, R2 changes from Ron to Roff
When Vc increases from 1.5V to 2V, R2 = Roff and R1 changes from Roff to Ron
At Vc = 1.5V, R1 = R2 = Roff

## Notes/Equations

1.  This SPDT_Dynamic switch model, as opposed to the SPDT_Static model, can be used to dynamically switch states in response to the input control voltage. The input impedance of the control voltage port is infinite. Whenever the control voltage is greater than 2.0V, the input is connected to the (c>2) output with a resistance equal to Ron; otherwise, there is a resistance of Roff Ohm between the two pins. Similarly,

whenever the control voltage is less than 1.0V, the input is connected to the (c<1) output with a resistance of Ron; otherwise, there is a resistance of Roff between the two pins. Note that when the control voltage is between 1.0V and 2.0V, the switch is open with respect to both outputs, simulating the break operation before making connection.

2. Capacitances, leakage currents, or other non-idealities are not included in this model.

3. This model is primarily meant for usage in the time domain (transient and circuit envelope) simulation with baseband control voltages. Due to the instantaneous, abrupt nature of the switching action, driving the control input with a carrier frequency input, in either harmonic balance or circuit envelope, should be avoided or done with careful consideration of the number of harmonics required and potential convergence problems.

# SPDT_Static (Single Pole Double Throw Switch, Static)

## Symbol
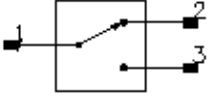


## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| State | State of switch: 0=off; 1=on | None | 1 |
| F1 | First frequency breakpoint | | |
| F2 | Second frequency breakpoint | | |
| F3 | Third frequency breakpoint | | |
| Loss1 | Attenuation for frequencies ≤ F1 | dB | 0 |
| Loss2 | Attenuation for F1< frequency ≤ F2 | dB | 0 |
| Loss3 | Attenuation for F2 > frequency ≤ F3 | dB | 0 |
| VSWR1 | Voltage standing wave ratio at both ports for frequencies ≤ F1 | None | 1 |
| VSWR2 | Voltage standing wave ratio at both ports for frequencies > F1, ≤ F2 | None | 1 |
| VSWR3 | Voltage standing wave ratio at both ports for frequencies > F2, ≤ F3 | None | 1 |
| Isolat | Isolation | dB | 100 |
| ZRef | reference impedance for all ports | Ohm | 50 |
| Temp | temperature | °C | None |
| CheckPassivity | check passivity flag: if set to yes, a passivity check is performed and a warning is output if the device is not passive; if set to no, a passivity check is not performed. | None | yes |

## Notes/Equations

1. The through path for each state of switch is defined as:

   | State | Through path |
   |-------|--------------|
   | 0 | pin 1 to pin 2 |
   | 1 | pin 1 to pin 3 |

2. This model is based on S-parameters (S12 = S21, S13 = S31).
3. Up to three frequency break points can be used to define four bands, with different losses and VSWR for each frequency band.
   Enter loss as a positive attenuation. In other words, a 0.5 dB loss is entered as Loss = 0.5, not Loss = −0.5.
4. Isolation is assumed to be constant across all frequency bands.
   When using this device in a transient simulation, the switch setting must be constant.

# SwitchV (Voltage Controlled Switch)

## Symbol
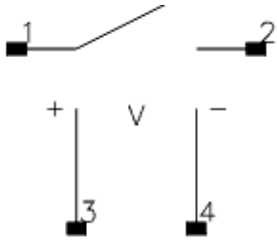


## Parameters

| Name | Description | Units | Default |
|---|---|---|---|
| Model | Model instance name | None | SWITCHVM1 |
| R1 | Resistance at voltage 1 | Ohm | 1.0 |
| V1 | Voltage 1 | V | 0.0 |
| R2 | Resistance at voltage 2 | MOhm | 1.0 |
| V2 | Voltage 2 | V | 1.0 |

## Range of Usage

R1, R2 > 0
V1 ≠ V2

## Notes/Equations

1. This component implements a voltage controlled switch. The switch resistance between nodes 1 and 2 varies as a function of the applied control voltage ($V_{CON}$ is the voltage between nodes 3 and 4):

$$R_{SW} = \begin{cases} R1 & \text{if} \quad V_{CON} \leq V1 \\ f(V_{CON}) & \text{if} \quad V1 \leq V_{CON} \leq V2 \\ R2 & \text{if} \quad V_{CON} \geq V2 \end{cases}$$

   where

$$f(V_{CON}) = \begin{cases} x = \dfrac{V_{CON} - V1}{V2 - V1} \\ if\, x < 1/2 \\ \quad y = 2x^2 \\ else \\ \quad y = 1 - 2(x-1)^2 \\ ln(R_{SW}) = \ln(R1) + y \times \ln(R2/R1) \end{cases}$$

For these equations, it is assumed that V1 < V2. If V2 < V1, swap R1 and V1 with R2 and V2 in the equation above.

VCON is defined so that f(V1) = R1 and f(V2) = R2 and provides a smooth transition between states. This is a plot of the switch resistance as a function of the control voltage for V1=1, R1=10, V2=3, R2=10000.

2. The Model name is optional. If the model name is not specified, then R1, V1, R2 and V2 must all be specified on the SwitchV instance; if the model name is specified, then R1, V1, R2 and V2 on the SwitchV instance are optional and override the values specified in the SwitchV_Model.

3. R1 and R2 are model parameters, not the resistance at nodes. Convergence problems may occur if the difference between R1 and R2 is too large.

4. V1 and V2 are model parameters, not node voltages. Convergence problems may occur if the difference between V1 and V2 is too small.

5. This component works in all analyses, including transient and circuit envelope simulation. If a SwitchV_Model item is not present, the SwitchV Model parameter should be blank.

6. *DC Operating Point Information* lists the DC operating point parameters that can be sent to the dataset.

| Name | Description | Units |
|---|---|---|
| Is | Current | A |
| Power | DC power dissipated | W |
| Vs | Voltage | V |

# SwitchV_Model (Voltage Controlled Switch Model)

## Symbol
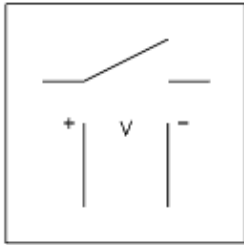


## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| R1 | Resistance at voltage 1 | Ohm | 1.0 |
| V1 | Voltage 1 | V | 0.0 |
| R2 | Resistance at voltage 2 | MOhm | 1.0 |
| V2 | Voltage 2 | V | 1.0 |
| AllParams | Data Access Component (DAC) Based Parameters | None | None |

## Range of Usage

R1, R2 > 0
V1 ≠ V2

## Notes/Equations

1. This model supplies values for a SwitchV component.
2. The range between R1 and R2 should not be too large, or convergence problems may occur.
3. The difference between V1 and V2 should not be too small, or convergence problems may occur.
4. R1, V1, R2 and V2 can be optionally specified on the SwitchV instance that uses this model; those values override the model values.
5. This model works in transient and circuit envelope simulation.
6. Use AllParams with a DataAccessComponent to specify file-based parameters (refer to the *DataAccessComponent* ). Note that model parameters that are explicitly specified take precedence over those specified via AllParams.

# VSum (Voltage Summer)

## Symbol
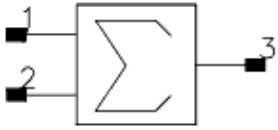


## Parameters

None

## Notes/Equations

1. This component has infinite input impedance and zero output impedance. Output voltage is equal to voltage summed from the two input ports.

# Tx/Rx Subsystems

- *RF PA CKT (RF Power Amplifier Circuit)* (ccsys)
- *RF RX SML (RF Receiver)* (ccsys)
- *RF TX SML (RF Transmitter)* (ccsys)

# RF_PA_CKT (RF Power Amplifier Circuit)

## Symbol
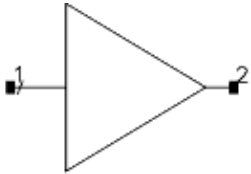


## Parameters

None

## Notes/Equations

1. This is a 2-stage BJT amplifier with the center frequency of 2GHz, small signal gain of ~30dB, and maximum power output of ~15dBm. This circuit is provided for the convenience of easy inclusion of a circuit level amplifier in an ADS Ptolemy Envelope cosimulation.

# RF_RX_SML (RF Receiver)

## Symbol



## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| RX_AntTemp | Receiving antenna noise temperature | None | 150 |
| RX_Gain | Receiver gain | dB | 50 |
| RX_NF | Receiver noise figure | dB | 5 |
| RF_Freq | RF frequency | MHz | 900 |
| RF_BW | RF bandwidth | MHz | 25 |
| IF_Freq1 | First IF frequency | MHz | 100 |
| IF_Freq2 | Second IF frequency | kHz | 400 |
| IF_BW | IF bandwidth | kHz | 30 |
| IP3in | Receiver input IP3 | dBm | −25 |
| RIn | Input resistance | Ohm | 50 |
| ROut | Output resistance | Ohm | 50 |

## Notes/Equations

1. This is a super-heterodyne RF receiver subsystem. Input is an RF signal; output is an IF signal. Three CW spurs are combined with the input RF signal to simulate the receiver interference performance.
   This subnetwork is provided for easy inclusion of an RF receiver in an ADS Ptolemy Envelope co-simulation.

# RF_TX_SML (RF Transmitter)

## Symbol

## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| IF_Freq | IF frequency | kHz | 400 |
| RF_Freq | RF frequency | MHz | 900 |
| RF_BW | RF bandwidth | kHz | 30 |
| TX_Gain | Transmitter gain | dB | 80 |
| PSat | Saturated power at output | dBm | 35 |
| RIn | Input resistance | Ohm | 50 |
| ROut | Output resistance | Ohm | 50 |

## Notes/Equations

1. This is an RF transmitter subsystem. Input is an IF signal; output is an RF signal. This subnetwork is provided for easy inclusion of an RF transmitter in an ADS Ptolemy Envelope co-simulation.

# System Data Models

The System - Data Models palette contains two types of components: those which enable the simulation-based extraction of behavioral profiles from circuit models and those which perform behavioral modeling using these profiles. There is an inherent pairing of these two types of components because parameters used to define behavioral profiles in data-based representation are specific to modeling theories and therefore data model requirements. Data-based behavioral modeling is an important tool for the *bottom-up* verification process; thus, each extractor-data model pair of this palette is also known as a verification model extraction (VME) pair. The most common usage of a VME pair during RFIC and MMIC design flows is as follows:

- Creation of a circuit level design with circuit level components and/or parametric behavioral components during the top-down design phase. Parameter-based behavioral models are located in the *Filters* , *System - Amps & Mixers* , and *System - Mod/Demod* palettes. These models are characterized by a few independent parameters such as frequency, power and load.
- Creation of a Harmonic Balance-based dataset or data file profiles of these circuit level models using the VME extractor components from the System - Data Models palette. Use the VME data model counterparts to recreate the behavior of the circuit level models. Compare circuit- and data-based behavioral models; calibrate the data-based behavioral models prior to hardware prototyping the circuit.
- Using measurement instruments to extract behavioral datasets or data files from hardware prototypes from the foundry. Using the calibrated VME data model to generate a hardware behavioral profile. Compare the hardware behavioral profile with the simulation-based behavioral profile to estimate the success of prototyping.

The *parameter-based behavioral models* typically provide superior speed whereas data-based behavioral models typically provide superior accuracy. The differences between *parameter* - and *data-based behavioral models* justify the palette emphasis on ADS flow (separate palettes for parametric and data-based models) rather than functionality (separate palettes for amplifiers, mixers, modulators each containing both parametric and data-based models).

The *BehavioralModels* example suite was developed to promote the usability of the *System-Data Models* components. This family of examples demonstrates the use and characteristics of most VME models. To access the example from the ADS Main window click on the following sequence of menu options: **File** > **Open** > **Example** > **BehavioralModels**. Notes for each example are available in the accompanying notes in *Behavioral Model Examples* (examples).

In addition to the data-based behavioral model components, the *System-Data Models* palette includes Balun3Port and Balun4Port; these components are also accessible from the *System-Passive* palette.

## Classification of ADS System Data Models

ADS provides data-based modeling for system-level VME components of the following

categories:

- Amplifiers: AmplifierH1H2, AmplifierP2D, AmplifierS2D, AmpLoadPull, VCA_Data
- Mixers: MixerHBdata, MixIMT_Data
- Modulators / Demodulators: IQ_Mod, IQ_Demod

Most data models accept customized dataset (*.ds) or data file (*.p2d, *.s2d, *.imt) inputs. These inputs are typically provided by data model extractors. Identified by the suffix Setup or setup, these extractors are prefabricated subcircuits that run a predetermined simulation with user-settable sweep values on the circuit level model of interest and report the input and output stimuli in the dataset or data file form. The user is encouraged to push into the extraction subcircuits and be familiar with the measurement metrics that are being reported to the dataset or data file.

Most system data models have their own data extractors; however, there are exceptions for the older models MixerIMT and MixerIMT2 for which data files must be manually generated or retrieved from a measurement instrument.

Data-Based Behavioral Modeling Components and Example Workspaces in ADS summarizes the availability and scope of each VME model currently available in ADS. This section describes component parameters and usage in further detail. For a discussion of the use of the ADS data model components from an application point of view please refer to the series of Application Notes being released under the title *Behavioral Modeling in ADS* .

**Data-Based Behavioral Modeling Components and Example Workspaces in ADS**

| Extractor Model | Data Model | Data Type | Comments on Scope of Modeling |
|---|---|---|---|
| AmpH1H2_Setup | AmpH1H2 | **.ds** | Models nonlinearity at odd and even harmonics using circuit-based data at fundamental and second harmonics. Refer to Examples > BehavioralModels > AmplH1H2_wrk |
| AmplifierP2D_Setup | AmplifierP2D | **.p2d** | Models nonlinearity and noise at fundamental frequency using circuit-based data at fundamental frequency. For circuit envelope simulations, modeling nonlinear distortion at fundamental frequency due to interference from spectral components within the envelope band. Refer to Examples > BehavioralModels > AmplifierP2D_wrk |
| AmplifierS2D_Setup | AmplifierS2D | **.s2d** | Models nonlinearity and noise at fundamental and odd-order harmonic frequencies using circuit-based data at fundamental frequency. Assumes odd-order polynomial fitting for harmonic calculations. Refer to Examples > BehavioralModels > AmplifierS2D_wrk |
| LoadPullSetup | AmpLoadPull | **.ds** | Models nonlinearity at fundamental frequency for various input and output impedance conditions using circuit-based data at single frequency for a range of source and load conditions. Refer to Examples > BehavioralModels > AmpLoadPull_wrk |
| VCA_Setup | VCA_Data | **.ds** | Models nonlinearity at fundamental frequency for various settings of control voltage using circuit-based data at single frequency for a range of bias conditions. Refer to Examples > BehavioralModels > VCA_wrk |
| MixIMTA_SetupMixIMTB_Setup | MixIMT_Data | **.imt** | Models nonlinearity at all mixing products for single (O-type and A-type) or multi (B-type) IMT files. Refer to *Examples > BehavioralModels > MixIMT_wrk* |
| MixerHBsetup | MixerHBdata | **.ds** | Models nonlinearity at fundamental, mixing product frequencies at the mixer output (IF) due to 1tone input signals (RF and LO) based on 1-tone harmonic balance characterization of circuit level mixer. Refer to Examples > BehavioralModels > MixerHBdata_wrk |
| (Manual extraction) | MixerIMT [††] | **.imt** [†] | Models nonlinearity at fundamentals and mixing product frequencies at the mixer output (IF) based on 1-tone input signals and a table of mixing coefficients. |
| (Manual extraction) | MixerIMT2 | **.imt** [†] | Models nonlinearity at fundamentals and mixing product frequencies at the mixer output (IF) based on 1-tone input signals and a table of mixing coefficients. |
| IQ_Mod_Setup | IQ_Mod_Data | **.ds** | Models large-signal in-phase and quadrature components during modulation Refer to Examples > BehavioralModels > IQ_Mod_wrk |
| IQ_Demod_Setup | IQ_Demod_Data | **.ds** | Models large-signal in-phase and quadrature components during demodulationRefer to Examples > BehavioralModels > IQ_Demod_wrk |
| [†]    This data model accepts only O-type IMT data. [††]    This is an obsolete component. | | | |

# Components

- *AmpH1H2_Setup (Amplifier, Fundamental and 2nd Harmonic vs. Input Power Setup)* (ccsys)
- *AmpH1H2 (Amplifier, Fundamental and 2nd Harmonic vs. Input Power)* (ccsys)
- *AmplifierP2D (P2D File Amplifier, FDD-Based, for Single Carrier Signal)* (ccsys)
- *AmplifierP2D Setup (Extractor Component for AmplifierP2D)* (ccsys)
- *AmplifierS2D (S2D File Amplifier, Polynominal Model for Nonlinearity)* (ccsys)
- *AmplifierS2D Setup (Extractor Component for AmplifierS2D)* (ccsys)
- *AmpLoadPull (SDD Load-Pull Amplifier)* (ccsys)
- *IQ Demod Data (IQ Demodulator Behavioral Model)* (ccsys)
- *IQ Demod Setup (IQ Demodulator Setup)* (ccsys)
- *IQ Mod Data (IQ Modulator Behavioral Model)* (ccsys)
- *IQ Mod Setup (IQ Modulator Setup)* (ccsys)
- *LoadPullSetup (Load Pull Setup)* (ccsys)
- *MixerHBdata (2-Tone HB Mixer)* (ccsys)
- *MixerHBsetup (2-Tone HB Mixer Setup)* (ccsys)
- *MixerIMT2 (Intermodulation Table Mixer)* (ccsys)
- *MixerIMT (Obsolete Intermodulation Table Mixer)* (ccsys)
- *MixIMTA Setup (Single-RF Intermodulation Table Extractor)* (ccsys)
- *MixIMTB Setup (Multi-RF Intermodulation Table Extractor)* (ccsys)
- *MixIMT Data (Multi-RF Intermodulation Table Mixer Data Model)* (ccsys)
- *VCA Data (Voltage Controlled Amplifier)* (ccsys)
- *VCA Setup (Voltage Controlled Amplifier Setup)* (ccsys)

# AmpH1H2_Setup (Amplifier/Fundamental and 2nd Harmonic vs. Input Power Setup)

## Symbol
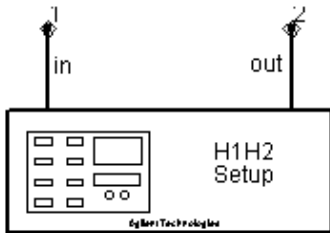


## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Freq | Fundamental frequency; must be a positive real number | GHz | 1.0 |
| Order | Order for Harmonic Balance analysis; must be an integer greater than 0 | None | 10 |
| Pin_Start | Start value of input power sweep for harmonic balance analysis; must be a positive real number | dBm | −50 |
| Pin_Stop | Stop value for input power sweep for harmonic balance analysis; must be a positive real number | dBm | −20 |
| Pin_Step | Interval for input power sweep during harmonic balance analysis; must be a positive real number | dB | 10 |

## Notes/Equations

1. AmpH1H2_Setup performs a swept input power Harmonic Balance simulation of a circuit-level amplifier and generates a dataset for subsequent use by the data-based system model AmpH1H2. The AmpH1H2_Setup extractor model is demonstrated in the example AmpH1H2_wrk.
2. AmpH1H2_Setup can be pushed into for a view of the implementation. If necessary, the component can be copied and modified to suit individual needs.
3. This component is capable of extracting circuit behavior at only one nominal frequency Freq. In order to characterize an amplifier over a range of frequencies, separate extractions must be performed and separate datasets generated at each frequency of interest.
4. In order for AmpH1H2 to produce an accurate model of the circuit-level amplifier characterized via AmpH1H2_Setup, the Order parameter for AmpH1H2_Setup must be large enough to prevent aliasing of higher-order frequency components. The recommended value is 5 for mildly non-linear circuits and 11-15 for highly non-linear circuits.
5. The linear input power sweep is defined by the parameters Pin_Start, Pin_Stop and Pin_Step are used directly by the underlying Harmonic Balance controller.
6. The extracted ADS dataset is assigned the name of the extraction design by default.

# AmpH1H2 (Amplifier/Fundamental and 2nd Harmonic vs. Input Power)

## Symbol
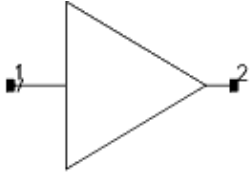


## Parameters

| Name | Description | Units | Default |
|---|---|---|---|
| Dataset | Name of dataset containing data for this amplifier model | None | dataset.ds |
| G1expr [†] | Gain expression for fundamental frequency | None | Vout[1] |
| G2expr [†] | Gain expression for second harmonic frequency | None | Vout[2] |
| SP11 [††] | Forward reflection coefficient | None | polar(0,0) |
| SP22 [††] | Reverse reflection coefficient | None | polar(0,180) |
| SP12 [††] | Reverse transmission coefficient | None | 0 |
| NF | Noise figure | dB | None |
| NFmin | Minimum noise figure at Sopt | dB | None |
| Sopt [††] | Optimum Source Reflection for Minimum Noise Figure, use x + j × y, polar(x,y), dbpolar(x,y) for complex value | None | None |
| Rn | Equivalent noise resistance; must be a non-negative real number | | |
| Z1 | Reference impedance for Port1 | | |
| Z2 | Reference impedance for Port2 | | |

[†] The gain parameters must be updated to appropriate gain expressions based on the MeasEqn expressions of AmpH1H2_Setup component which was used to create the dataset. Refer to example AmpH1H2_wrk. [††] These parameters can be reported in any of the following complex number formats: x + j × y, polar(x,y), dbpolar(x,y), vswrpolar(x,y)

## Notes/Equations

1. AmpH1H2 is a data-based system model of a circuit-level amplifier. The circuit-level amplifier is characterized by a dataset generated by the extractor component AmpH1H2_Setup. Various examples of the use of AmpH1H2 are provided in the example AmpH1H2_wrk.

2. This amplifier model does not require the explicit specification of nominal frequency at which non-linear modeling is to be performed. Although technically it can be used at any frequency during simulation, the dataset file contains the nonlinear profile at the single frequency point defined on the AmpH1H2_Setup component during

extraction. Hence, effectively the AmpH1H2 model is confined to reproducing behavior accurately at that frequency only. Since the AmpH1H2 model does not have a built-in warning mechanism to detect discrepancies between simulation and extraction frequencies, the user of this model needs explicit apriori knowledge of the extraction frequency during behavioral simulation.

3. AmpH1H2 models both odd- and even-order harmonics at the amplifier output. However, its reliance on the fundamental and second-order harmonic expressions G1expr and G2expr. respectively, guarantee high accuracy only for these two spectral components. For details of the modeling accuracy of various harmonics in a 1-tone swept power Harmonic Balance simulation, refer to the data display plot behavioral_level_amp_SOITOI.dds in the example AmpH1H2_wrk.

4. The dataset contains information about the forward transmission characteristic S21. The other S-parameters S11, S12 and S22, can be set explicitly on the data model instance. Likewise, noise parameters NF, NFmin, Sopt and Rn, can be set explicitly using relevant parameters on the model instance.

5. It is important to note that although port impedances Z1 and Z2 can be set to arbitrary values during behavioral simulation, the dataset is typically extracted at port impedance values of 50 Ohms. Ideally, the settings for Z1 and Z2 inside the subcircuit of the extractor model AmpH1H2_Setup should be set up as complex conjugates of the input and output impedances, respectively, of the circuit amplifier. The resulting dataset would then contain the behavior under matched source and load conditions. In order to translate the dataset into the behavior of AmpH1H2, the AmpH1H2 Z1 and Z2 parameters must then be set to the complex conjugates of AmpH1H2_Setup component's Z1 and Z2 values, i.e. to the original values of the circuit amplifiers port impedances.

# AmplifierP2D (P2D File Amplifier, FDD-Based, for Single Carrier Signal)

## Symbol
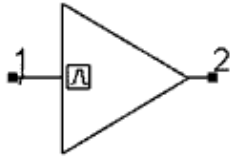


## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Freq | Fundamental frequency | GHz | 1.0 |
| P2DFile | Filename for P2D data | None | p2dfile.p2d |
| FilteringOption † | CE simulation option of pre / post filtering envelope frequencies relative to nonlinearity; enumerated options are: NoFilter, PreFilter, PostFilter | None | NoFilter |
| CEFreqSpacing † | Sampling frequency used for envelope simulation | MHz | 1.0 |
| iVarN †† | Name of Nth multi-dimensional variables N=(1,...,N). Use if present in P2D file | None | None |
| iValN †† | Value of Nth multi-dimensional variables N=(1,...,N). Use real or integer values only | None | None |
| ImpNoncausalLength | Non-causal function impulse response order | Integer | None |
| ImpMode | Convolution mode | Integer | None |
| ImpMaxFreq | Maximum frequency to which device is evaluated |  |  |
| ImpMaxOrder | Maximum allowed impulse response order | Integer | None |
| ImpWindow | Smoothing window | Integer | None |
| ImpRelTol | Relative impulse response truncation factor | None | None |
| ImpAbsTol | Absolute impulse response truncation factor | None | None |

† The FilteringOption and CEFreqSpacing parameters are provided (beginning with 2003C) to enhance circuit envelope simulation capabilities; refer to note 8 and note 10 for details. The default values suppress effects of FilteringOption and CEFreqSpacing parameters to ensure backwards compatibility. CEFreqSpacing is the equivalent of ImpDeltaFreq in

† † AmplifierS2D with the restriction that it must always have an assigned value in AmplifierP2D. If multi-dimensional VAR statements are present in the P2DFile, then these variables are required to be listed in iVarN as indicated in note 12 and corresponding values assigned to them using the iValN parameter.

## Notes/Equations

1. AmplifierP2D is a data-based system model of a circuit-level amplifier. The circuit-level amplifier is characterized by a P2D file generated either by the extractor component AmplifierP2D_Setup in a simulation environment or by a measurement instrument such as a network analyzer. Various examples of the use of the AmplifierP2D data model are provided in the example AmpP2D_wrk.
2. A P2D file, named *.p2d, contains small- and large-signal 2-port S-parameter data with optional noise parameters and intermodulation table data. The AmplifierP2D

model only uses data related to amplifier modeling; it ignores frequency translation and intermodulation table data that may be contained in a generic P2D file. The referenced P2D file should reside in the data subdirectory of the current workspace. For details on P2D file format, refer to *P2D Format* (cktsim).

3. The AmplifierP2D model blocks DC. For CE simulations, baseband signals are blocked.

4. For small-signal simulations such as AC and S-parameter analyses, AmplifierP2D only uses the small-signal S-parameters of the P2D file. For S-parameter analysis, the nominal frequency must be set as Freq=freq, where freq is the ambient simulation frequency. If the Freq parameter value falls outside the range of frequencies in the small-signal section of the P2D file, an error message is reported in the simulator log window and the is simulation terminated. Linear interpolation of S-parameters is performed within the small-signal range to emulate behavior at a frequency not explicitly registered in the data file. Linear extrapolation is enabled but not recommended outside the small-signal frequency range because modeling accuracy cannot be guaranteed outside the data points of the P2D file.

5. For large-signal frequency analysis such as those based on Harmonic Balance, the power-dependent S-parameters from the ACDATA block are used in conjunction with small-signal parameters. For these simulations, Freq must be set to the fundamental frequency explicitly or through the pre-defined variable _freq1. Setting Freq=freq causes errors. For an N-tone Harmonic Balance simulation any of N frequencies specified on the controller may be chosen as nominal frequency for the AmplifierP2D component by setting Freq=_freqX, where X is in the integer range [1, N].

6. If the Freq parameter value falls outside the range of frequencies in the large-signal section of the P2D file, a warning message is reported in the simulator log window although the simulation is allowed to proceed. Linear interpolation along the frequency axis is permitted within the small- and large-signal range but extrapolation is not advised on either side of the large-signal limits because of insufficient data. It is important to scan the simulator log window at the end of any large-signal simulation involving the AmplifierP2D component to ensure that frequency limits were not exceeded.

7. If the power incident at the input of the amplifier model during large-signal frequency domain analyses, such as Harmonic Balance, LSSP or Circuit Envelope simulations, exceeds the maximum value of P1 in the P2D file a warning message is sent the simulator log window concerning unsupported modeling at high drive levels.

8. The FilteringOption parameter allows the user to regulate modeling of distortion effects at the fundamental frequency due to other frequencies within the envelope bandwidth during a circuit envelope simulation. Some harmonics and intermods produced by the non-fundamental tones within the CE-band contribute to the distortion of the fundamental tone at the output of the amplifier for high drive levels at the input. In the design of complex circuit level amplifiers this distortion is either regulated at the amplifier input or output using a narrowband filter. In order to imitate this feature during behavioral simulation AmplifierP2D enables the user to set the FilteringOption parameter to PreFilter or PostFilter so that appropriate filtering effects can be modeled without changing the P2D data file. It is useful to perform such CE-band distortion evaluation for certain applications (communications systems, for example). The NoFilter default setting suppresses estimation of envelope bandwidth distortion at the nominal frequency Freq; this setting is sufficient for all non-CE simulations.

9. The PreFilter or PostFilter option is for use during circuit envelope simulations only. Prefiltering of non-fundamental frequencies reduces the potential of distortion close to power saturation of the fundamental frequency at the output; postfiltering allows maximal non-linear distortion to occur at amplifier output. Maximally flat filtering is

assumed across the entire envelope bandwidth. This distortion modeling feature is highlighted in the example design BEH_P2D_CE_filter.

10. The CEFreqSpacing parameter is used to select the granularity of envelope band frequencies that are allowed to impact the distortion of the fundamental frequency at amplifier output. There is an inherent performance trade-off in using this parameter: the smaller its value the more accurate the simulation but the longer the duration of the CE simulation. If filtering is enabled and CEFreqSpacing exceeds half the value of Freq a warning message is reported to the simulation log window and simulation proceeds without the use of filtering due to violation of the Nyquist sampling requirement for CEFreqSpacing. The recommended value for CEFreqSpacing is $10^{-3}$ (value assigned to Freq). This feature is highlighted in the example design BEH_P2D_CE_sample.

11. AmplifierP2D is implemented using the FDD model. Unlike the SML models Amplifier and Amplifier2 or the VME model AmplifierS2D, AmplifierP2D does not produce harmonics or intermods in a Harmonic Balance simulation.
In a Harmonic Balance analysis, the only frequency component that can pass through AmplifierP2D is that specified by Freq. Signals at all other frequencies will see a ground at the input of AmplifierP2D. However, in a Circuit Envelope simulation, the envelope signal around the carrier frequency Freq will pass through AmplifierP2D and cause distortion at the fundamental Freq, whereas all signals outside the envelope band are ignored.
To simulate multi-tone behavior involving harmonic modeling, consider using the AmpH1H2 or AmplifierS2D models.

12. The response of AmplifierP2D can be user-selected from a body of multi-dimensional P2D data by specifying the names and ambient values of such variables in the iVarN and iValN entries prior to behavioral simulation. Details of multi-dimensional P2D file generation are discussed in the notes for AmplifierP2D_Setup component and the format of these files is described in *Working with Data Files* (cktsim). If a given multi-dimensional P2D file contains "V" VAR variables then the names of all these variables need to be assigned using the iVar1 through iVarV parameters of the AmplifierP2D instance. Although the ordering of these names can be arbitrary relative to those in the P2D file, the existence and ordering of values in the iVal1 through iValV parameters should always correspond to that of iVar1 through iVarV. If a multi-dimensional VAR contained in the P2D data file is not specified on the iVarN of an AmplifierP2D instance reading that file, the simulation will be terminated on due to insufficient data for multi-dimensional data indexing. Currently, only real and integer values of multidimensional iVarN are supported.

# AmplifierP2D_Setup (Extractor Component for AmplifierP2D)

## Symbol
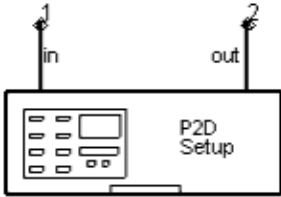


## Parameters

| Name | Description | Unit | Default |
|------|-------------|------|---------|
| Filename | Name of P2D file to be generated by extractor | None | p2dfile.p2d |
| Order | Order for Harmonic Balance simulation inside extractor; must be an integer greater than 0 | None | 10 |
| Freq_Start | Start value for sweep of large-signal frequencies; must be a positive real number | GHz | 1.0 |
| Freq_Stop | Stop value for sweep of large-signal frequencies; must be a positive real number | GHz | 2.0 |
| Freq_Step [†] | Interval for linear sweep of large-signal frequencies; must be a positive real number | GHz | 0.1 |
| Pin_Start | Start value for sweep of large-signal input power; must be a real number | dBm | −10.0 |
| Pin_Stop | Stop value for sweep of large-signal input power; must be a real number | dBm | 15.0 |
| Pin_Step [†] | Interval for linear sweep of large-signal input power; must be a positive real number | dB | 5.0 |
| SSFreq_Start [††] | Start value for sweep of small-signal frequencies; must be a positive real number if specified | | |
| SSFreq_Stop [††] | Stop value for sweep of small-signal frequencies; must be a positive real number if specified | | |
| SSFreq_Step [†] , [††] | Interval for linear sweep of small-signal frequencies; must be a positive real number if specified | | |

† Only linear sweeps using intervals of Freq_Step, Pin_Step, SSFreq_Step are permitted frequency and power extractions using the AmplifierP2D_extractor. †† Small-signal frequency settings, if left unspecified, will default to large-signal frequency specifications. The default behavior of this component is to enforce SSFreq_Start <= Freq_Start, SSFreq_Stop >= Freq_Stop and SSFreq_Step <= Freq_Step. Violations of this will result in internal correction with warning messages being reported to the simulator log window. Refer to note 4 for modifying this default behavior.

## Notes/Equations

1. AmplifierP2D_Setup is a data file extractor component used to create multi-dimensional behavioral profiles of a circuit level amplifier in the P2D format. The use of this component is highlighted in the design CKT_P2D_extraction of the example *AmpP2D_wrk.*

2. AmplifierP2D_Setup can be pushed into for a view of the implementation. If necessary, the component can be copied and modified to suit individual needs.

3. AmplifierP2D_Setup performs a linear swept input frequency over linear swept input power P2D (Harmonic Balance) simulation of a circuit-level amplifier and generates a P2D file for subsequent use by the data-based system model AmplifierP2D. The *Freq_* and *Pin_* parameters regulate the large-signal frequency and input power sweeps, respectively.

4. The small-signal frequency sweep can be specified independently of the large- signal frequency sweep using *SSFreq_* parameters. When unspecified, these values internally default to large-signal values. When specified in violation of the rule that SSFreq_Start <= Freq_Start, SSFreq_Stop >= Freq_Stop and SSFreq_Step <= Freq_Step, this rule is imposed internally during simulation and warning messages appear in the simulator log window. This ensures the small-signal frequency sweep spans the large-signal frequency sweep and is at least as finely grained. This is the coarsest granularity for reliable modeling using the AmplifierP2D data model.

5. In order for AmplifierP2D to produce an accurate model of the circuit-level amplifier characterized via AmplifierP2D_Setup, the Order parameter for AmplifierP2D_Setup must be large enough to prevent aliasing of higher-order frequency components. Because the AmplifierP2D data model reads and operates primarily on its assigned fundamental frequency, it is important to provide an accurate estimate of the fundamental response at the circuit level amplifier output. Therefore, it is important to use a large order of harmonics in AmplifierP2D_Setup to reconstruct the fundamental accurately during data modeling. The recommended value is 5 for mildly non-linear circuits and 11-15 for highly non-linear circuits.

6. For information regarding the P2D data format, refer to *PDF Format* (cktsim). Multi-dimensional P2D files may be generated by simply placing a chain of *ParamSweep* controllers above the AmplifierP2D_Setup instance and assigning the lowest sweep to have SimInstance[1]="Xi1.HB" where "Xi" is the instance name of the P2D extractor and HB1 is the default name of the P2D controller within its subcircuit.

# AmplifierS2D (S2D File Amplifier, Polynominal Model for Nonlinearity)

**Symbols:**
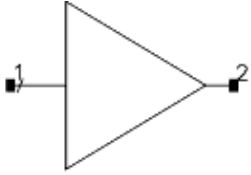


**Parameters**

| Name | Description | Units | Default |
|---|---|---|---|
| S2DFile [†] | Filename for S2D data; enumerated as: S2D filename, File Based | None | s2dfile.s2d |
| SSFreq | Small signal frequency for interpolating S-parameters; enumerated as: auto, freq, _freq1, _freq2, _freq3 | None | auto [††] |
| InterpMode | Interpolation mode; enumerated as: Linear, Cubic Spline, Cubic, Value Lookup, .... , Value | None | Linear |
| InterpDom | Interpolation geometry; enumerated as: Data Based, Rectangular, Polar, DB | None | Data Based |
| GCFreq [†††] | Reference frequency for gain compression; must be a positive real number if specified | | |
| VarName [‡] | Name of 1st multi-dimensional variable; use if present in S2D file | None | None |
| VarValue [‡] | Value of 1st multi-dimensional variable; use real or integer values only | Integer | None |
| iVarN [‡] | Name of Nth multi-dimensional variables N={2,...,N}; use if present in S2D file | None | None |
| iValN [‡] | Value of Nth multi-dimensional variables N={2,...,N}; use real or integer values only | Integer | None |
| ImpNoncausalLength [‡‡] | Non-causal function impulse response order | Integer | None |
| ImpMode [‡‡] | Convolution mode | Integer | None |
| ImpMaxFreq [‡‡] | Maximum frequency to which device is evaluated | | |
| ImpDeltaFreq [‡‡] | Sample spacing in frequency | Integer | |
| ImpMaxOrder [‡‡] | Maximum allowed impulse response order | Integer | None |
| ImpWindow [‡‡] | Smoothing window | None | None |
| ImpRelTol [‡‡] | Relative impulse response truncation factor | None | None |
| ImpAbsTol [‡] | Absolute impulse response truncation factor | Integer | None |

[†] The "S2D Filename" option allows the entry of an S2D file directly. The "File Based" option enables the use of a DataAccessComponent (DAC) for reading in non-S2D file based data.

[††] Implies "_freq1" for non-DC part of HB/CE/LSSP simulations. It implies "freq" for all other simulations. For multitone simulations setting SSFreq=_freq1 explicitly gives better results. In general "_freqX" allows the specification of any one of the X tones in an X-tone Harmonic Balance simulation, e.g. selecting SSFreq=_freq2 sets the small-signal frequency of interest to be the second tone of a 2+ tone HB simulation.

[†††] If manually specified, this frequency must explicitly exist in the S2D file. No interpolation or extrapolation is done along frequency axis for estimating compression characteristics.

[‡] If multi-dimensional VAR statements are present in the S2DFile, then these variables are required to be listed in iVarN as indicated in note 8] and corresponding values assigned to them using the iValN parameter. Note that VarName and VarValue serve as iVar1 and iVal1 respectively because of historical reasons.

[‡‡] These parameters are also used as specifications for the Amplifier and Amplifier2 components.

### Range of Usage

S2D gain compression valid parameter ranges:
GCOMP3: IP3 > 1DBC + 10.6

GCOMP4: IP3 > PS + 8.6
GCOMP5: PS > 1DBC + 3
GCOMP6: PS > 1DBC + 3, IP3 > 1DBC + 10.6

### Notes/Equations

1.  AmplifierS2D is a data-based behavioral model that can represent fundamental tone(s), odd-order harmonics and odd-order intermodulation products of a circuit level amplifier. This model accepts S2D data profiles which can be generated using the AmplifierS2D_Setup component as shown in the design *CKT_S2D_extraction* in the example *AmpS2D_wrk* . Various examples of the use of AmplifierS2D are also provided in this workspace.

2.  An S2D file, named * . *s2d* , contains small-signal 2-port S-parameter data with optional noise parameters. In addition, it contains small-signal S-parameter data blocks, noise data blocks, and one of seven possible gain compression blocks {GCOMP1, ... , GCOMP7}. For information regarding the small- or large-signal S-parameter *.s2d* data format, refer to *S2D Format* (cktsim). The AmplifierS2D model only uses data related to amplifier modeling; it ignores frequency translation data that may be contained in a generic S2D file. The referenced S2D file should reside in the data subdirectory of the current workspace. The AmplifierS2D_Setup extractor model can be used to automatically generate S2D files.

3.  The AmplifierS2D model blocks DC. For CE simulations, baseband signals are blocked.

4.  For small signal simulations such as AC and S-parameter analyses, AmplifierS2D uses only the small signal ACDATA section of the S2D file. For small signal simulations *SSFreq* is effectively set to *freq* internally regardless of user defined enumeration. The *GCFreq* parameter has no effect on small signal simulations and may be left unassigned.
    For large signal simulations, *SSFreq* may be defined using any of the five possible enumerations with the following effects:
    *   *SSFreq = auto* implies first HB/Envelope tone. This implies that the reference value for small signal S-parameters is fixed at the 2- port values for *_freq1* regardless of the operational compression frequency *GCFreq* . This is the desirable setting for single tone HB/Envelope simulations.
    *   *SSFreq = freq* implies that the amplifier will be sensitive to small signal values based on individual input tones. This setting is desirable for multi-tone HB / Envelope analyses.
    *   *SSFreq = _freqN, N={1,2,3},* provides a special provision allowing you to select the small signal value at the first, second, or third HB/Envelope tone on file as the reference for all incoming tones in a generic HB simulation.

    > ⓘ **Note** Leaving the default value of *SSFreq=auto* enables the AmplifierS2D instance to function adequately and conveniently for small signal SP/AC as well as single tone HB/Envelope simulations. For multi-tone large signal simulations, *SSFreq* should be set to *freq* to allow small signal frequency sensitivity.

5.  For large-signal frequency analysis such as those based on Harmonic Balance, the small-signal S-parameters are used in conjunction with gain compression information in the GCOMPx block. Only odd-order harmonics are modeled during such large-signal simulations because information about even-order harmonics cannot be derived from the GCOMPx specification.

6.  If gain compression information is presented in the GCOMP7 format at multiple

frequencies, the AmplifierS2D *GCFreq* parameter must be assigned to one of the explicitly defined large-signal frequencies of the * .s2d file. The data model does not interpolate across large-signal frequencies so any violation of this requirement will result in termination of the simulation. This model resolves conflicts between user-assigned *GCFreq* values and large-signal frequency points in GCOMP7-based S2D files as shown in *Using GCFreq to Resolve GCOMP7 Frequency Conflicts*.

**Using GCFreq to Resolve GCOMP7 Frequency Conflicts**

| Number of GCOMP7 blocks | Large signal frequencies (s2dfreq) specified in S2D file? | GCFreq specified on AmplifierS2D? | GCFreq = one s2dfreq? | Simulation Result. Compression data used at ... |
|---|---|---|---|---|
| 0 | N/A | any | N/A | N/A |
| 1 | no | no | N/A | simulation frequency |
| | | yes | N/A | GCFreq |
| | yes | no | N/A | s2dfreq |
| | | yes | no | N/A. Simulation terminated with error message. |
| | | | yes | GCFreq = s2dfreq |
| > 1 | no | no | N/A | simulation frequency |
| | | yes | N/A | GCFreq. Use first available power sweep from GCOMP7 |
| | yes | no | N/A | N/A. Simulation terminated with error message. |
| | | yes | no | N/A. Simulation terminated with error message. |
| | | | yes | GCFreq = s2dfreq |

7. For good performance using AmplifierS2D, include multiple GCOMP7 blocks at closely-spaced frequencies.
8. AmplifierS2D can access multi-dimensional S2D files as shown in BEH_S2D_HB_1tone of the example AmpS2D_wrk. Previous versions of this component required a *DataAccessComponent* (DAC) for accessing multi-dimensional data. A DAC is no longer necessary to access S2D data from a single file. Please note that multiple S2D files cannot be fed to the same AmplifierS2D component in the course of a single simulation. When S2D data is available in multiple files, consolidate the S2D data into one file and use this master file as the input to AmplifierS2D. Relevant sections of this new file can now be conveniently accessed using the VarName, VarValue and iVar2, iVal2 through iVar10 and iVal10 parameters of the component. Automatic multi-dimensional S2D file generation is facilitated by using the AmplifierS2D_Setup component.
9. The S21 values expressed in the GCOMP7 block are not absolute values of S21 at the various input power PIN levels. They are differential values in dB domain between power-swept S-parameters at the frequency of the GCOMP7 block and the small-signal S-parameters at the same frequency in the ACDATA block of the *.s2d file. The formats in which ACDATA and GCOMP7 data are represented in the same file may differ and may be in any of the forms DB / MA / RI. When calculating GCOMP7 values

manually, exercise care to convert absolute values of both large- and small-signal S-parameters to the dB domain before subtracting the small-signal S-parameters from the large-signal S-parameters to obtain the numeric values in the GCOMP7 section.

10. AmplifierS2D is implemented using the SML Amplifier model. However, the gain compression information available via an S2D file is limited to the forward transmission characteristic defined by S21. If an application requires the knowledge and use of all four 2-port S-parameter variations but not the use of harmonics, consider using the AmplifierP2D model. If even-order harmonics must be modeled but sweeps of frequency are not required, consider using the AmpH1H2 model instead.

# AmplifierS2D_Setup (Extractor Component for AmplifierS2D)

## Symbol
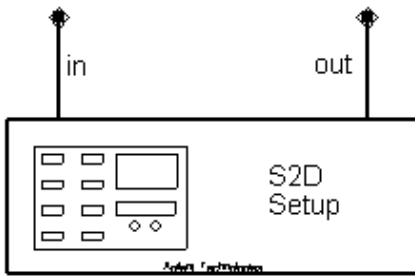


## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Filename | Name of P2D file to be generated by extractor | None | s2dfile.s2d |
| Order | Order for Harmonic Balance simulation inside extractor; must be an integer greater than 0 | None | 10 |
| Freq_Start | Start value for sweep of large-signal frequencies; must be a positive real number | GHz | 1.0 |
| Freq_Stop | Stop value for sweep of large-signal frequencies; must be a positive real number | GHz | 2.0 |
| Freq_Step [†] | Interval for linear sweep of large-signal frequencies; must be a positive real number | GHz | 0.1 |
| Pin_Start | Start value for sweep of large-signal input power; must be a real number | dBm | −10.0 |
| Pin_Stop | Stop value for sweep of large-signal input power; must be a real number | dBm | 15.0 |
| Pin_Step [†] | Interval for linear sweep of large-signal input power; must be a positive real number | dB | 5.0 |
| SSFreq_Start [††] | Start value for sweep of small-signal frequencies; must be a positive real number if specified | | |
| SSFreq_Stop [††] | Stop value for sweep of small-signal frequencies; must be a positive real number if specified | | |
| SSFreq_Step [†] , [††] | Interval for linear sweep of small-signal frequencies; must be a positive real number if specified | | |

† Only linear sweeps using intervals of Freq_Step, Pin_Step, SSFreq_Step are permitted frequency and power extractions using the AmplifierP2D_extractor.†† Small-signal frequency settings, if left unspecified, will default to large-signal frequency specifications. The default behavior of this component is to enforce SSFreq_Start <= Freq_Start, SSFreq_Stop >= Freq_Stop and SSFreq_Step <= Freq_Step. Violations of this will result in internal correction with warning messages being reported to the simulator log window. Refer to note 4 for information about modifying this default behavior.

## Notes/Equations

1. AmplifierS2D_Setup is a data file extractor component used to create multi-dimensional behavioral profiles of a circuit level amplifier in the S2D GCOMP7 format. The use of this component is highlighted in the design CKT_S2D_extraction of the example *AmpS2D_wrk.*
2. AmplifierS2D_Setup can be pushed into for a view of the implementation. If necessary, the component can be copied and modified to suit individual needs.
3. AmplifierS2D_Setup performs a linear swept input frequency over linear swept input power P2D (Harmonic Balance) simulation of a circuit-level amplifier and generates an S2D file for subsequent use by the data-based system model AmplifierP2D. The *Freq_* and *Pin_* parameters regulate the large-signal frequency and input power sweeps, respectively. The power sweep is registered in the GCOMP7 format.
4. The small-signal frequency sweep can be specified independently of the large-signal frequency sweep using *SSFreq_* parameters. When unspecified, these values internally default to large-signal values. When specified in violation of the rule that *SSFreq_Start <= Freq_Start, SSFreq_Stop >= Freq_Stop* and *SSFreq_Step <= Freq_Step* , this rule is imposed internally during simulation and warning messages appear in the simulator log window. This ensures the small-signal frequency sweep spans the large-signal frequency sweep and is at least as finely grained. This is the coarsest granularity for reliable modeling using the AmplifierS2D data model.

5. In order for AmplifierS2D to produce an accurate model of the circuit-level amplifier characterized via AmplifierS2D_Setup, the Order parameter for AmplifierS2D_Setup must be large enough to prevent aliasing of higher-order frequency components. Because the AmplifierS2D data model reads and operates primarily on its assigned fundamental frequency, and uses this frequency to create odd-order harmonics at the output, it is important to provide an accurate estimate of the fundamental response at the circuit level amplifier output. Therefore, it is important to use a large order of harmonics in AmplifierS2D_Setup to reconstruct the fundamental accurately during data modeling. The recommended value is 5 for mildly non-linear circuits and 11-15 for highly non-linear circuits.

6. For information regarding the S2D data format, refer to the *S2D Format* (cktsim). Multi-dimensional S2D files may be generated by simply placing a chain of *ParamSweep* controllers above the AmplifierS2D_Setup instance and assigning the lowest sweep to have *SimInstance[1]="Xi.HB1"* where " *Xi* " is the instance name of the S2D extractor and HB1 is the default name of the P2D controller within its subcircuit.

# AmpLoadPull (SDD Load-Pull Amplifier)

## Symbol
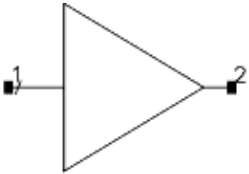
## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Dataset † | Name of dataset containing load-pull data for this amplifier model | None | dataset.ds |
| † File-based data is also supported via use of the DataAccessComponent | | | |

## Notes/Equations

1. For amplifier designers, the AmpLoadPull and LoadPullSetup components address the issue of output match in one or more frequency bands. This is often investigated via load-pull contours indicating the load impedances, which, when presented to the output of an amplifier with a given source impedance and power, cause a certain power to be delivered to the load. The LoadPullSetup component extracts a dataset, given ranges and steps for input power, output reflection coefficient magnitude and output reflection coefficient angle. AmpLoadPull uses this dataset and allows fast behavioral amplifier simulations for all input power and output reflection coefficient values in the specified ranges.

2. This amplifier model uses the load-pull ADS dataset generated by the LoadPullSetup component. This behavioral profile can be referenced using the Dataset parameter of the AmpLoadPull component.

3. AmpLoadPull is useful in modeling nonlinear amplifiers with or without input/output matching networks. This model can be used in all types of simulations.

4. When using this model in other simulations, ensure the input power and the load value are within the range of the original data.

5. The use of this device and its extractor component is highlighted in the example *AmpLoadPull_wrk.*

6. This model does not generate harmonics, and its performance is not affected by the source or load impedances presented to it at the harmonic frequencies.

# IQ_Demod_Data (IQ Demodulator Behavioral Model)

## Symbol
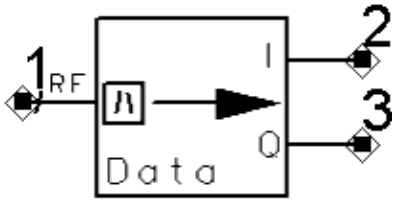


## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Dataset | Name of dataset generated by IQ_Demod_Setup | None | dataset.ds |
| Freq | Carrier Frequency | GHz | 1.0 |

## Notes/Equations

1. IQ_Demod_Data is a data-based system model of a circuit-level I/Q demodulator. The circuit-level demodulator is characterized by a dataset generated by the system component IQ_Demod_Setup. The use of this component is highlighted in the example *IQ_Demod_wrk.*

2. The demodulation behaviors are extracted for a single carrier frequency. Only the demodulation distortions are characterized, so it is possible to set the frequency parameter in IQ_Demod_Data to differ from the value set in IQ_Demod_Setup. The demodulation will occur at the frequency set in IQ_Demod_Data. The extracted distortions to the baseband signal as it passes through IQ_Demod_Data will be applied at this frequency. IQ_Demod_Data does not model any dispersion in these distortions across the baseband bandwidth.

3. IQ_Demod_Data is designed for Circuit Envelope system verification where various filters typically eliminate unwanted frequency components. Therefore, IQ_Demod_Data makes no attempt to model harmonic components. The signals generated on the I and Q output ports are purely baseband signals. IQ_Demod_Data does not generate any frequency components at any carrier frequency.

4. In order for IQ_Demod_Data to produce an accurate model of the circuit-level demodulator characterized via IQ_Demod_Setup, the IQ_Demod_Setup Order parameter must be large enough to prevent aliasing of higher-order frequency components. For mildly non-linear circuits, an Order of about 5 should suffice, while an Order in the range of 10-15 is recommended for highly non-linear circuits. Providing this criterion is met, the Order parameter for the simulation controller can be very low when using IQ_Demod_Data in a Circuit Envelope simulation. Typically, an Order of around 3 should suffice for accurate demodulation modeling. Note, however, that accurate modeling of other components of the complete system may necessitate a larger value for this parameter.

5. For a circuit-level demodulator, the impedances presented by the input and output

pins will generally be a complicated function of all state variables within the demodulator. For the IQ_Demod_Data, a certain simplification is necessary as no information about the environment in which the demodulator will later reside is known at the time of extraction. The input impedance at the RF input pin is extracted at the RF carrier frequency specified in IQ_Demod_Setup, and is a function of the input power. The impedances at the I and Q output pins are extracted at DC, and are modeled as constant impedances.

6. The extrapolation properties of IQ_Demod_Data above Pin-Stop are very poor. When using IQ_Demod_Data, ensure that the RF input signal does not exceed the *Pin_Stop* value set in IQ_Demod_Setup when the model was extracted. Extrapolation to signal levels below the *Pin_Start* value set in IQ_Demod_Setup will generally be good as long as the *Pin_Start* value lies within the linear operating range of the demodulator.

7. IQ_Demod_Data does not model noise.

# IQ_Demod_Setup (IQ Demodulator Setup)

## Symbol



## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Freq | Modulation carrier frequency; must be a non-negative real number | GHz | 1.0 |
| Order | Order for Harmonic Balance analysis; must be a positive integer greater than 0 | None | 10 |
| Pin_Start [†] | Start value of input power sweep for harmonic balance analysis; must be a real number | dBm | −40 |
| Pin_Stop [†] | Stop value for input power sweep for harmonic balance analysis; must be a real number | dBm | 10 |
| Pin_Step [†] | Interval for input power sweep during harmonic balance analysis; must be a real number | dB | 2 |

[†] All power values are referenced to a 50 Ohm system. Pin_Stop >= the following values:10 + 20 × log10 (vmax) where vmax is the maximum peak voltage magnitude of the modulated signal.Pmax,z + 10×log10(z/50) where Pmax,z, is the maximum modulated signal power in dBm for a z Ohm system.

## Notes/Equations

1. IQ_Demod_Setup performs a swept Harmonic Balance simulation of a circuit-level I/Q demodulator and generates a dataset for subsequent use by the data-based system model IQ_Demod_Data.The use of this component is highlighted in the example *IQ_Demod_wrk.*
2. In order for IQ_Demod_Data to produce an accurate model of the circuit-level demodulator characterized by IQ_Demod_Setup, the Order parameter for IQ_Demod_Setup must be large enough to prevent aliasing of higher-order frequency components. For mildly non-linear circuits, an Order of approximately 5 should suffice, while an Order in the range of 10-15 is recommended for highly non-linear circuits.
3. The demodulation behaviors are extracted for a single carrier frequency and over the range of modulated signal power specified by *Pin_Start* and *Pin_Stop* . These power values are specified in dBm with respect to a 50 Ohm system. For an accurate IQ_Demod_Data behavioral model, Pin_Stop must be set at least as large as the

expected maximum modulated signal power level. Refer to the Parameters table footnotes for conversion if working in other than a 50-Ohm system.

4. Push into IQ_Demod_Setup in a Schematic window for a view of the swept Harmonic Balance simulation controllers. This component can be copied and modified to suit individual needs; typically, this would entail changing the Harmonic Balance controller *HB1* to achieve more efficient simulations.For more information, refer to *Harmonic Balance Simulation* (cktsimhb).

5. The extracted ADS dataset is assigned the name of the extraction design by default.

# IQ_Mod_Data (IQ Modulator Behavioral Model)

## Symbol
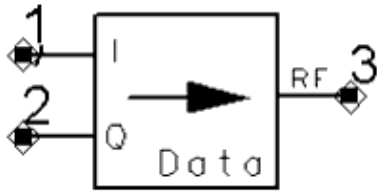


## Parameters

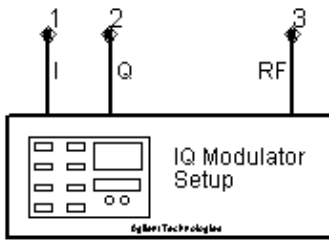| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Dataset | Name of dataset generated by IQ_Mod_Setup | None | dataset.ds |
| Freq | Carrier Frequency | GHz | 1.0 |

## Notes/Equations

1. IQ_Mod_Data is a data-based system model of a circuit-level I/Q modulator. The circuit-level modulator is characterized by a dataset generated by the system component IQ_Mod_Setup.The use of this component is highlighted in the example *IQ_Mod_wrk.*

2. The modulation behaviors are extracted for a single carrier frequency. Only the modulation distortions are characterized, so it is possible to set the frequency parameter in IQ_Mod_Data to differ from the value set in IQ_Mod_Setup. The modulation will occur at the frequency set in IQ_Mod_Data. The extracted distortions to the baseband signal as it passes through IQ_Mod_Data will be applied at this frequency. IQ_Mod_Data does not model any dispersion in these distortions across the baseband bandwidth.

3. IQ_Mod_Data is designed for Circuit Envelope system verification where various filters typically eliminate unwanted frequency components. Therefore, IQ_Demod_Data makes no attempt to model harmonic components. The signal generated on RF output port is only a baseband signal modulated onto the carrier frequency. IQ_Mod_Data does not generate any frequency components at DC or at any harmonic of the carrier frequency.

4. In order for IQ_Mod_Data to produce an accurate model of the circuit-level demodulator characterized via IQ_Mod_Setup, the Order parameter for IQ_Mod_Setup must be large enough to prevent aliasing of higher-order frequency components. For mildly non-linear circuits, an Order of approximately 5 should suffice, while an Order in the range of 10-15 is recommended for highly non-linear circuits. Providing this criterion is met, the Order parameter for the simulation controller can be very low when using IQ_Mod_Data in a Circuit Envelope simulation. Typically, an Order of approximately 3 should suffice for accurate demodulation modeling. Note, however, that accurate modeling of other components of the complete system may necessitate a larger value for this parameter.

5. For a circuit-level modulator, the impedances presented by the input and output pins will generally be a complicated function of all state variables within the modulator. For the IQ_Mod_Data, a certain simplification is necessary as no information about the environment in which the modulator will later reside is known at the time of extraction. Specifically, IQ_Mod_Data presents a constant inward-looking impedance at the input and output pins. The input impedances at the baseband input pins are extracted at DC, while the impedance at the RF output pin is extracted at the RF carrier frequency specified in IQ_Mod_Setup.

6. The extrapolation properties of IQ_Mod_Data above *Pin-Stop* are very poor. When using IQ_Mod_Data, please ensure that the input baseband signal does not exceed the Pin_Stop value set in IQ_Mod_Setup when the model was extracted. Note that the *Pin_Stop* value is in reference to the complex baseband signal, which is typically a factor of sqrt(2), or 3 dB, larger than the power of the signal at either individual input pin. Extrapolation to signal levels below the *Pin_Start* value set in IQ_Mod_Setup will generally be good as long as the *Pin_Start* value lies within the linear operating range of the modulator.

7. IQ_Mod_Data does not model noise.

# IQ_Mod_Setup (IQ Modulator Setup)

## Symbol



## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Freq | Modulation carrier frequency; must be a non-negative real number | GHz | 1.0 |
| Order | Order for Harmonic Balance analysis; must be a positive integer greater than 0 | None | 10 |
| Pin_Start [†] | Start value of input power sweep for harmonic balance analysis; must be a real number | dBm | −40 |
| Pin_Stop [†] | Stop value for input power sweep for harmonic balance analysis; must be a real number | dBm | 10 |
| Pin_Step [†] | Interval for input power sweep during harmonic balance analysis; must be a real number | dB | 2 |

[†] All power values are referenced to a 50-Ohm system. Pin_Stop >= the following values:$10 + 20 \times \log 10$ (vmax) where vmax is the maximum peak voltage magnitude of the modulated signal sqrt($|Vi|^2 + |Vq|^2$ ).Pmax,z + $10 \times \log 10(z/50)$ where Pmax,z, is the maximum modulated signal power in dBm for a z Ohm system.

## Notes/Equations

1. IQ_Mod_Setup performs a swept Harmonic Balance simulation of a circuit-level I/Q modulator and generates a dataset for subsequent use by the data-based system model IQ_Mod_Data.The use of this component is highlighted in the example *IQ_Mod_wrk* .

2. In order for IQ_Mod_Data to produce an accurate model of the circuit-level demodulator characterized by IQ_Mod_Setup, the Order parameter for IQ_Mod_Setup must be large enough to prevent aliasing of higher-order frequency components. For mildly non-linear circuits, an Order of approximately 5 should suffice, while an Order in the range of 10-15 is recommended for highly non-linear circuits.

3. The modulation behaviors are extracted for a single carrier frequency and over the range of modulation signal power specified by *Pin_Start* and *Pin_Stop* . These power values are specified in dBm with respect to a 50-Ohm system. For an accurate

IQ_Mod_Data behavioral model, *Pin_Stop* must be set at least as large as the expected maximum modulated signal power level. Refer to the Parameters table footnotes for conversion if working in other than a 50-Ohm system.

4. Push into IQ_Mod_Setup in a Schematic window for a view of the swept Harmonic Balance simulation controllers. This component can be copied and modified to suit individual needs; typically, this would entail changing the Harmonic Balance controller *HB1* to achieve more efficient simulations. For more information, refer to *Harmonic Balance Simulation* (cktsimhb).

5. The extracted ADS dataset is assigned the name of the extraction design by default.

# LoadPullSetup (Load Pull Setup)

## Symbol
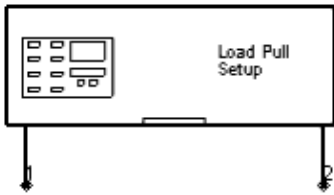


## Parameters

| Name | Description | Units | Default |
|---|---|---|---|
| Freq | Frequency of extraction; must be a non-negative real number | GHz | 1.0 |
| Order | Order for Harmonic Balance simulation inside extractor; must be an integer greater than 0 | None | 10 |
| Pin_Start | Start value for sweep of large-signal input power; must be a real number | dBm | −50 |
| Pin_Stop | Stop value for sweep of large-signal input power; must be a real number | dBm | −20 |
| Pin_Step | Interval for linear sweep of large-signal input power; must be a positive real number | dB | 10 |
| GamAng_Start | Start value for angle of output reflection coefficient; must be a real number | deg | −180 |
| GamAng_Stop | Stop value for angle of output reflection coefficient; must be a real number | deg | 180 |
| GamAng_Step † | Interval for angle of output reflection coefficient; must be a real number | deg | 20 |
| GamMag_Start | Start value for magnitude of output reflection coefficient; must be a non-negative real number | None | 0.1 |
| GamMag_Stop | Stop value for magnitude of output reflection coefficient; must be a non-negative real number | None | 0.9 |
| GamMag_Step † | Interval for magnitude of output reflection coefficient; must be a non-negative real number | None | 0.1 |
| † Only linear sweeps using intervals of GamAng_Step, GamMag_Step are permitted using LoadAmpPull. | | | |

## Notes/Equations

1. For amplifier designers, the AmpLoadPull and LoadPullSetup components address the issue of output match in one or more frequency bands. This is often investigated via load-pull contours indicating the load impedances that, when presented to the output of an amplifier with a given source impedance and power, cause a certain power to be delivered to the load. The LoadPullSetup component extracts an ADS dataset, given ranges and steps for input power, output reflection coefficient magnitude and output reflection coefficient angle. AmpLoadPull uses this dataset and allows fast behavioral amplifier simulations for all input power and output reflection coefficient values in the specified ranges.The use of this component is highlighted in the

example *AmpLoadPull_wrk.*

2. Although the behavioral data collected via LoadPullSetup is found using Harmonic Balance simulations, the subsequent behavioral amplifier simulations are not restricted to Harmonic Balance. In fact, the behavioral amplifier is assumed to be used for subsequent system verification (BER, ACPR etc. for Tx, Rx) within a Circuit Envelope framework. In such applications, various filters typically eliminate unwanted DC and harmonic components. Therefore, the load-pull suite was not designed to predict such components. This means that

   - for a 1-tone Harmonic Balance simulation at frequency *f* , only the frequency component at *f* (fundamental) is retained. All others will be invalid or zero as they are assumed to be filtered later.
   - for a 2-tone Harmonic Balance simulation at frequencies f1 and f2 (practically very close), only the frequency components at f1 and f2 (fundamentals) and those at 2 × f1-f2 and 2 × f2-f1 (intermodulation) are retained as these will be very close and cannot be assumed to be filtered. All others will be invalid or zero as they are assumed to be filtered later.

3. LoadPullSetup can be pushed into for a view of the implementation. If necessary, the component can be copied and modified to suit individual needs.

4. The extracted ADS dataset is assigned the name of the extraction design by default.

# MixerHBdata (2-Tone HB Mixer)

## Symbol
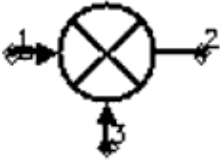


## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Dataset | Name of dataset for 2-tone harmonic balance mixer generated by MixerHBsetup | None | dataset.ds |
| RFindexExpr [†] | RF tone mixing index defined in the MixerHBsetup component at the time of dataset extraction; typically set to "Mix(2)" | None | Mix(2) |
| LOindexExpr [†] | LO tone mixing index defined in the MixerHBsetup component at the time of dataset extraction; typically set to "Mix(1)" | None | Mix(1) |
| IFwaveExpr [†] | IF wave variable defined in the MixerHBsetup component at the time of dataset extraction; typically set to "b2" | None | b2 |
| IMdata [††] | Intermodulation data flag: can be −1 or 1 only | None | −1 |
| ConvGain [†††] | Conversion gain multiplier from RF port (Port 1) to IF port (Port 2) | None | dbpolar(0,0) |
| SP11 [†††] | RF port reflection coefficient | None | polar(0,0) |
| SP12 [†††] | IF port to RF port leakage coefficient | None | polar(0,0) |
| SP13 [†††] | LO port to RF port leakage coefficient | None | polar(0,0) |
| SP21 [†††, ‡] | RF port to IF port transmission/leakage coefficient | None | polar(0,0) |
| SP22 [†††] | IF port reflection coefficient | None | polar(0,180) |
| SP23 [†††, ‡] | LO port to IF port transmission/leakage coefficient | None | polar(0,0) |
| SP31 [†††] | RF port to LO port leakage coefficient | None | polar(0,0) |
| SP32 [†††] | IF port to LO port leakage coefficient | None | polar(0,0) |
| SP33 [†††] | LO port reflection coefficient | None | 0 |
| NF | Input double side band noise figure | dB | None |
| NFmin | Minimum double sideband noise figure at Sopt | dB | None |
| Sopt [†††] | Optimum Source Reflection for Minimum Noise Figure, use x+j*y, polar(x,y), dbpolar(x,y) for complex value | None | None |
| Rn | Equivalent noise resistance; must be a non-negative real number | | |
| Z1 | Reference impedance for RF port | | |
| Z2 | Reference impedance for IF port | | |
| | | | |

| Z3 | Reference impedance for LO port | | |
|---|---|---|---|
| P_RFnom ‡‡ | Power incident at the RF port; must be manually synchronized with source signal value at this port | dBm | −50 |
| P_LOnom ‡‡ | Power incident at the LO port; must be manually synchronized with source signal value at this port | dBm | −5 |
| M_RF ‡‡ | Number of RF harmonics to be considered from dataset | None | 4 |
| N_LO ‡‡ | Number of LO harmonics to be considered from dataset | None | 4 |

† Typically not edited by user. Should be synchronized with expressions registered in dataset created using MixerHBsetup.

† † Set to -1for lower sideband or 1 for upper sideband. Refer to note 5 for details.

† † † These parameters can be reported in any of the following complex number formats: x + j*y, polar(x,y), dbpolar(x,y), vswrpolar(x,y)

‡ Transmission/leakage implies that the value of this parameter is used as an additive supplement to gain expressions between RF->IF and LO->IF ports.

‡‡ Refer to note 7 for details.

**Notes/Equations**

1. MixerHBdata is a data-based system model of a circuit-level mixer. The dataset used by MixerHBdata is generated from the circuit-level mixer by the behavioral extractor component MixerHBsetup (available in the System-Data Models palette). Ports 1, 2 and 3 on MixerHBdata correspond to the RF, IF and LO ports of MixerHBsetup.
2. The extraction process using MixerHBsetup is done at fixed user-supplied RF and LO power levels. MixerHBdata performs linear interpolation along either power axes if the data model-based verification process is done at a different RF or LO power level. Therefore, it is recommended that the data model be used either in a known linear region of operation or cautiously within a narrow input power range if it is necessary to replicate non-linear operation.
3. MixerHBdata can be used in Harmonic Balance, Circuit Envelope, and Transient simulations.
4. MixerHBdata is currently configured to produce IF output with zero phase regardless of the IF output phase represented in the MixerHBsetup extracted dataset.
5. A dataset extracted using the MixerHBsetup component contains both upper and lower sideband harmonics and intermodulation products. By formal definition the upper sideband contains frequencies above the higher of the two mixing frequencies. For example, in a typical RFIC receiver, the center frequency is the *RFfreq* because LOfreq < RFfreq. In the MixerHBdata component the *IMdata* parameter is used to prompt the use of only one of these two bands for deriving an internal polynomial model for frequency response. Once this model has been constructed, it is used to represent behavior of both sidebands in the MixerHBdata simulation output.
6. If a 2-tone Harmonic Balance simulation is to be performed on a MixerHBdata component using Freq[1]=LOfreq and Freq[2]=RFfreq in the simulator component, then setting IMdata=-1 selects the lower sideband frequencies and IMdata=1 selects the upper sideband frequencies from the dataset. The harmonics and intermodulation products are defined as |m × RFfreq - n × LOfreq| in the lower sideband and |m × RFfreq + n × LOfreq| in the upper sideband, where *m* and *n* are integers whose absolute values are bounded by _M_RF_ and _N_LO_ (the maximum harmonic order

parameters of the MixerHBdata component).

7. By the convention that -M_RF ≤ m ≤ M_RF and -N_LO ≤ n ≤ N_LO, and the assumption that M_RF and N_LO are of comparable magnitudes, the difference frequencies |m × RFfreq - n × LOfreq| are guaranteed to be lower than RFfreq and the summed frequencies |m × RFfreq + n × LOfreq| are guaranteed to be higher than RFfreq. Thus, the IMdata parameter helps in sideband selection and should be assigned values of -1 or 1 only. For proper modeling _M_RF_ and _N_LO_ of the MixerHBdata component should be set to the value of RForder and LOorder settings of the MixerHBsetup that was used to extract the behavioral dataset.

8. The use of this component is highlighted in the example *MixerHBdata_wrk*.

# MixerHBsetup (2-Tone HB Mixer Setup)

## Symbol
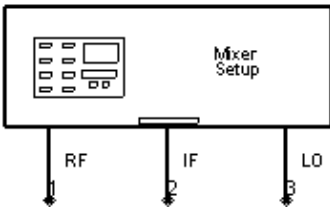


## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| RFfreq | RF frequency. Denotes incoming message signal whether it is intended for up or down conversion; must be a non-negative real number | GHz | 1 |
| LOfreq | LO frequency. Denotes frequency of local oscillator or carrier frequency; must be a non-negative real number | GHz | 1.1 |
| RFpwr | Power incident at the RF port; must be manually synchronized with source signal value at this port | dBm | −20 |
| LOpwr | Power incident at the LO port; must be manually synchronized with source signal value at this port | dBm | 0 |
| RForder | Number of RF harmonics allowed to participating in mixing; must be a positive integer greater than 0 | None | 10 |
| LOorder | Number of LO harmonics allowed to participating in mixing; must be a positive integer greater than 0 | None | 10 |

## Notes/Equations

1. MixerHBsetup extracts the behavior of a circuit-level mixer using a Harmonic Balance simulation. The extracted dataset can be subsequently used by the corresponding data-based system model MixerHBdata. Ports RF, IF and LO ports of the MixerHBsetup component correspond to the ports 1, 2 and 3 on the MixerHBdata component.

2. MixerHBsetup can be pushed into for a view of the extractor implementation including access to the built-in Harmonic Balance simulator component. Note that equations and variables of the extractor subcircuit are referenced by name from parameters of the MixerHBdata component. Any user-initiated changes made to a specific instance of MixerHBsetup must be reflected in a complementary instance of MixerHBdata for correct dataset indexing for behavioral modeling.

3. The use of this component is highlighted in the example *MixerHBdata_wrk* located at `$HPEESOF_DIR/examples/BehavioralModels`.

4. The extracted ADS dataset is assigned the name of the extraction design by default.

# MixerIMT2 (Intermodulation Table Mixer)

**Symbols:**
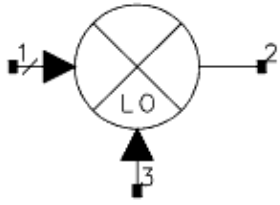


**Parameters**

| Name | Description | Units | Default |
|---|---|---|---|
| ConvGain [†] | Conversion gain multiplier from RF port (Port 1) to IF port (Port 2) | None | dbpolar(0,0) |
| SP11 [†] | RF port reflection coefficient | None | polar(0,0) |
| SP12 [†] | IF port to RF port leakage coefficient | None | polar(0,0) |
| SP13 [†] | LO port to RF port leakage coefficient | None | polar(0,0) |
| SP21 [†] | RF port to IF port transmission/leakage [††] coefficient | None | polar(0,0) |
| SP22 [†] | IF port reflection coefficient | None | polar(0,180) |
| SP23 [†] | LO port to IF port transmission/leakage [††] coefficient | None | polar(0,0) |
| SP31 [†] | RF port to LO port leakage coefficient | None | polar(0,0) |
| SP32 [†] | IF port to LO port leakage coefficient | None | polar(0,0) |
| SP33 [†] | LO port reflection coefficient | None | 0 |
| NF | Input double side band noise figure; must be a non-negative real number | dB | None |
| NFmin | Minimum double sideband noise figure at Sopt; must be a positive real number | dB | None |
| Sopt [†] | Optimum Source Reflection for Minimum Noise Figure, use x + j × y, polar(x,y), dbpolar(x,y) for complex value | None | None |
| Rn | Equivalent noise resistance; must be a non-negative real number | | |
| R1 | Reference impedance for RF port; must be a non-negative real number | | |
| R2 | Reference impedance for IF port; must be a non-negative real number | | |
| R3 | Reference impedance for LO port; must be a non-negative real number | | |
| M_RF [†††] | IMT order for RF port | None | 15 |
| N_LO [†††] | IMT order for LO port | None | 15 |
| IMTValueType | IMT value type: enumerated as {dB, dBm} | None | dB |
| IMT_File | File containing intermodulation table; accepted formats MDIF: IMT, P2D, S2D | None | imtfile.imt |

[†] These parameters can be reported in any of the following complex number formats: x + j × y, polar(x,y), dbpolar(x,y), vswrpolar(x,y)

[††] Transmission/leakage implies that the value of this parameter is used as an additive supplement to gain expressions between RF->IF and LO->IF ports.

[†††] Must be set to a value less than or equal to the number of harmonics represented in the IMT table. The limit on the number of rows and columns in an IMT table is 15.

## Notes/Equations

1. For new designs, this component should be used only if MixIMT_Data does not provide necessary functionality.
2. To use *NFmin* , *Sopt* , and *Rn* to characterize noise, the following condition must be satisfied for a realistic model:

$$\frac{Rn}{Zo} \geq \frac{To(Fmin-1)|1+Sopt|^2}{T\,4} \frac{(1-|S_{11}|^2)}{|1-Sopt\,S_{11}|^2}$$

If the noise parameters attempt to describe a system that requires negative noise

(due to *Rn* being too small), the negative part of the noise will be set to 0 and a warning message will be issued.

3. Use the function polar(mag,ang), or dbpolar(dB,ang), or vswrpolar(VSWR,ang) to convert these specifications into a complex number.

4. For an S-parameter or a noise figure sinusoidal ripple, use the function ripple (mag, intercept, period, variable); for example ripple(0-1, 0-1, 10 MHz, Freq).
   Example: S21=dbpolar(10+ripple( ),0.)

5. MixerIMT2 accepts only O-type IMT files where only single RF on single LO mixing can be specified with implicit assumption about reference RF and LO frequencies. These files are inherently single side banded descriptions of mixer behavior and do not contain IF phase information. For details on various IMT file formats see *Working with Data Files* (cktsim) > IMT Format.

6. When supplied a signal at frequency *Freq_Signal* at the RF port and a local oscillator frequency *Freq_LO* at the LO port, MixerIMT2 produces intermodulation products at all frequencies
   $|m \times Freq\_LO +/- n \times Freq\_Signal|$
   where
   $-M\_RF \le m \le M\_RF$ and $-N\_LO \le n \le -N\_LO$
   The simulation controller may be set to compute fewer frequency components than shown in the above equation. As a general rule, the values of _M_RF_ and _N_LO_ should be set to the order(s) of RF and LO tones on the simulator component and these values must be less than or equal to the number of rows and columns of the IMT table. The largest IMT table supported by this component is of size $16 \times 16$.
   If there are multiple frequencies at the RF and LO ports, there will be a superposition of the RF-LO intermodulation tones. For example, given the frequencies $LO_1$ and $LO_2$ at the LO port, and $Signal_1$ and $Signal_2$ at the RF port, there will be intermodulation products at the frequencies:
   $|m \times Freq\_LO_1 +/- n \times Freq\_Signal_1|$,
   $|m \times Freq\_LO_1 +/- n \times Freq\_Signal_2|$,
   $|m \times Freq\_LO_2 +/- n \times Freq\_Signal_1|$,
   $|m \times Freq\_LO_2 +/- n \times Freq\_Signal_2|$
   where
   $-M\_RF \le m \le M\_RF$ and $-N\_LO \le n \le \_N\_LO$
   Note that there will be no products at $| LO 1 +/- LO 2|$, $| Signal 1 +/- Signal 2|$ and other RF or LO self-modulation products at the output of the mixer.

7. MixerIMT2 can be used in all simulations except for Frequency Converting AC analysis.

8. MixerIMT2 leakage terms are specified in two separate ways: the IMT file entries, and the *SPij* parameters. For example, LO to IF leakage is given by the (M=1, N=0) term in the IMT file as well as the SP23 parameter. When both specifications are given, the total leakage is given by the complex sum of the two specifications. It is generally advisable to leave SP23=SP21=0; in this case, the LO to IF and the RF to IF leakages are specified by the IMT file.

# MixerIMT (Obsolete Intermodulation Table Mixer)

**Symbols:**
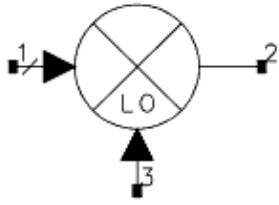


**Parameters**

| Name | Description | Units | Default |
|---|---|---|---|
| SS_SideBand | Produce UPPER or LOWER sideband at output port for linear analysis. | None | UPPER |
| ConvGain † | Conversion gain multiplier from RF port (Port 1) to IF port (Port 2). | None | dbpolar(0,0) |
| SP11 † | RF port reflection coefficient. | None | polar(0,0) |
| SP22 † | IF port reflection coefficient. | None | polar(0,180) |
| SP33 † | LO port reflection coefficient. | None | 0 |
| NF | Input double side band noise figure; must be a non-negative real number. | dB | None |
| NFmin | Minimum double sideband noise figure at Sopt; must be a positive real number. | dB | None |
| Sopt † | Optimum source reflection for minimum noise figure; must have magnitude less than 1.0. | | |
| Rn | Equivalent noise resistance; must be a non-negative real number. | Ohm | |
| R1 | Reference impedance for RF port; must be a non-negative real number. | Ohm | |
| R2 | Reference impedance for IF port; must be a non-negative real number. | Ohm | |
| R3 | Reference impedance for LO port; must be a non-negative real number. | Ohm | |
| InThresh | Voltage threshold for signal input at RF port; must be a positive real number. | V | |
| LoThresh | Voltage threshold for signal input at LO port; must be a positive real number. | V | |
| IMT_File | File containing intermodulation table. Accepted formats MDIF: IMT, P2D, S2D. | | imtfile.imt |

† These parameters can be reported in any of the following complex number formats: x + j*y, polar(x,y), dbpolar(x,y), vswrpolar(x,y)

**Notes/Equations**

1. This component is included for the convenience of customers who have existing designs with the MixerIMT model. New instances can be accessed by typing MixerIMT into the Component History field in the Schematic window, pressing Enter, and

moving your cursor to the drawing area to place it. For new designs, this component should be used only if MixIMT_Data does not provide necessary functionality.

2. MixerIMT does not function in Transient simulations; MixerIMT blocks baseband signals in Circuit Envelope simulations; MixerIMT does not produce intermodulation among RF input tones (refer to *note 9*). Replace MixerIMT with MixerIMT2 for these functions.

3. The default value of S12=0 to avoid creating cyclic interdependencies between the RF and IF ports.

4. If no IMT file is referenced, the output (IF) signal is the product of the input RF and LO spectral tones only.

5. If *NFmin* , *Sopt* , and *Rn* are used to characterize noise, the following criteria must be met for a realistic model:

$$\frac{Rn}{Zo} \geq \frac{To(Fmin-1)|1+Sopt|^2}{T\,4} \frac{(1-|S_{11}|^2)}{|1-Sopt\,S_{11}|^2}$$

*Rn* will be reset to a value that meets this criteria if it does not satisfy this condition.

6. Use the function polar(mag,ang), or dbpolar(dB,ang), or VSWRpolar(VSWR, ang) to convert these specifications into a complex number.

7. For an S-parameter or a noise figure sinusoidal ripple, use the function ripple (mag, intercept, period, variable); for example ripple(0-1, 0-1, 10 MHz, Freq).
   Example: S21=dbpolar(10+ripple( ),0.)

8. This model blocks dc.

9. MixerIMT accepts only O-type IMT files where only single RF on single LO mixing can be specified with implicit assumption about reference RF and LO frequencies. These files are inherently single side banded descriptions of mixer behavior and do not contain IF phase information. For details on various IMT file formats see *Working with Data Files* (cktsim) > IMT Format. From such a file, MixerIMT produces intermodulation products at frequencies

```
|m &times; LO\+/- n &times; Signal|
```

where

```
-M &le; m  &le; M and -N &le; n &le; N
```

Also, *M* and *N* are the orders for LO and RF signal frequencies set in the Harmonic Balance or Circuit Envelope simulation controllers.
If there are multiple frequencies at the RF and LO ports, there will be a superposition of the RF-LO intermodulation tones. For example, given the frequencies *LO* 1 and *LO* 2 at the LO port, and *Signal* 1 and *Signal* 2 at the RF port, there will be intermodulation products at the frequencies:

$|m \times LO_1 +/- n \times Signal_1|$, $|m \times LO_1 +/- n \times Signal_2|$,

$|m \times LO_2 +/- n \times Signal_1|$, $|m \times LO_2 +/- n \times Signal_2|$

where

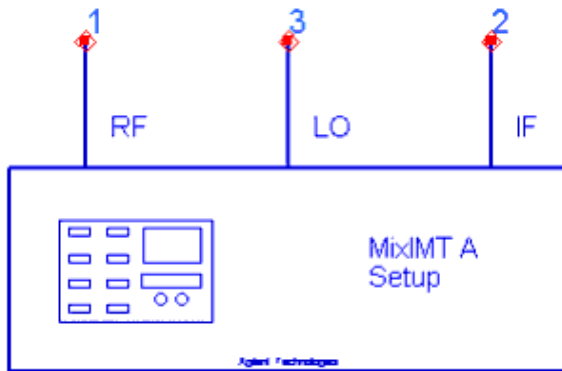-M_RF ≤ *m* ≤ M_RF and - N_LO_ ≤ *n* ≤ N_LO
There will be no products at | *LO* 1 +/- *LO* 2|, | *Signal* 1 +/- *Signal* 2| or any other RF or LO self-modulation products.

# MixIMTA_Setup (Single-RF Intermodulation Table Extractor)

## Symbol



## Parameters

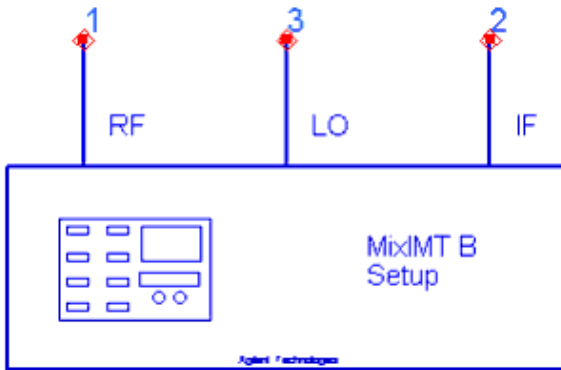| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Filename | Name of IMT file to be generated | | "imtfile.imt" |
| RFharms | Number of RF harmonics involved in mixing | None | 2 |
| LOharms | Number of LO harmonics involved in mixing | None | 2 |
| LOpowr_Start | Nominal or lowest power of LO signal generated | dBm | -10 |
| LOpowr_Stop [†] | Highest power of LO signal generated (if swept) | dBm | |
| LOpowr_Step [†] | Increment in LO signal power (if swept) | dB | |
| RFpowr_Start | Nominal or lowest power of RF signal generated | dBm | -50 |
| RFpowr_Stop [†] | Highest power of RF signal generated (if swept) | dBm | |
| RFpowr_Step [†] | Increment in RF signal power (if swept) | dB | |
| LOfreq_Start | Nominal or lowest frequency of LO signal generated | Hz | 1.0 GHz |
| LOfreq_Stop [†] | Highest frequency of LO signal generated (if swept) | Hz | |
| LOfreq_Step [†] | Increment in LO signal frequency (if swept) | Hz | |
| RFfreq_Start | Nominal or lowest frequency of RF signal generated | Hz | 1.1 GHz |
| RFfreq_Stop [†] | Highest frequency of RF signal generated (if swept) | Hz | |
| RFfreq_Step [†] | Increment in RF signal frequency (if swept) | Hz | |
| [†]  Values may be specified simultaneously in *_Stop and *_Step parameters, to sweep a corresponding variable. If both are left unspecified, only single point extraction occurs using the nominal value of the *_Start parameter. | | | |

**Notes/Equations**

1. MixIMTA_Setup is an intermodulation table generating component. When linked to a three pin mixer subcircuit, it is capable of stimulating the RF and LO inputs of the mixer using single-tone sinusoidal RF and LO signals at specified frequencies/powers, while capturing the IF output of the mixer to an intermodulation table (IMT) of specified size. It is possible to capture multiple IM tables to the same data file by sweeping RF and LO signal powers/frequencies.

2. A-type (single RF tone) IMT files contain double side banded complex IF voltage values (dBm,deg), in matrices sized ( 2* *RFharms* + 1 ) x ( *LOharms* + 1), where *RFharms* and *LOharms* number of harmonics participate in the mixing process. Details of file format are described in *Working with Data Files* (cktsim) > IMT Format.

3. Examine the sub-circuit of the extractor to see how the underlying harmonic balance simulation is set-up to achieve mixing. Although there is no theoretical upper limit on the number of harmonic indices that can be specified on this extractor, the user should establish a cut-off on the number of IF spurs to improve the performance of the underlying harmonic balance simulation.

4. To understand the use model for this extractor see example design: BehavioralModels > MixIMT_wrk > CKT_IMTA_extraction. Also see related component *MixIMTB_Setup (Multi-RF Intermodulation Table Extractor)* (ccsys).

# MixIMTB_Setup (Multi-RF Intermodulation Table Extractor)

## Symbol



## Parameters

| Name | Description | Units | Default |
|---|---|---|---|
| Filename | Name of IMT file to be generated | | "imtfile.imt" |
| RFharms[r] [†][†] | Number of harmonics of the r-th RF input involved in mixing | None | 1, r=1 |
| RFfreq[r] [†][†] | Fundamental frequency of r-th RF tone | Hz | 1.1 GHz, r=1 |
| RFpowr[r\] [†][†] | Power of r-th RF tone | dBm | -50, r=1 |
| LOharms | Number of LO harmonics involved in mixing | None | 2 |
| LOfreq_Start | Nominal or lowest frequency of LO signal generated | Hz | 1.0 GHz |
| LOfreq_Stop [††] | Highest frequency of LO signal generated (if swept) | Hz | |
| LOfreq_Step [††] | Increment in LO signal frequency (if swept) | Hz | |
| LOpowr_Start | Nominal or lowest power of LO signal generated | dBm | -10 |
| LOpowr_Stop [††] | Highest power of LO signal generated (if swept) | dBm | |
| LOpowr_Step [††] | Increment in LO signal power (if swept) | dB | |

[†] Add or remove elements of repeated parameters from dialog box. Ensure that identical number of harmonics, frequencies and powers are specified at all times. [††] Values may be specified simultaneously in *_Stop and *_Step parameters, to sweep a corresponding variable. If both are left unspecified, only single point extraction occurs using the nominal value of the *_Start parameter

## Notes/Equations

1. MixIMTB_Setup is an intermodulation table generating component. When linked to a three pin mixer subcircuit, it is capable of stimulating the RF and LO inputs of the mixer using multi-tone sinusoidal RF and LO signals at specified frequencies/powers,

while capturing the IF output of the mixer to an intermodulation table (IMT) of specified size. It is possible to capture multiple IM tables to the same data file by sweeping LO signal powers/frequencies.

2. B-type (multi-RF tone) IMT files contain double side banded complex IF voltage values (dBm,deg), in matrices sized [( 2* *RFharms[1]* + 1 )*..... *( 2* *RFharms[r]* + 1 )] x ( *LOharms* + 1) where *RFharms* various RF tones and *LOharms* number of harmonics participate in the mixing process. Details of file format are described in *Working with Data Files* (cktsim) > IMT Format.

3. Although there is no theoretical upper limit on the number of harmonic indices that can be specified on this extractor, the user should establish a cut-off on the number of IF spurs to improve the performance of the underlying harmonic balance simulation. Using the equation for matrix size defined above, it can be verified that a simple three-tone RF input, such as used for adjacent channel power rejection (ACPR) tests, mixed at RFharm[] = {3,2,2} with LOharm = 5 will generate a 175 x 5 IMT matrix. Lowering the spur precision by 1 on each RF tone will produce a more manageable 45 x 5 IMT matrix.

4. To understand the use model for this extractor see example design: BehavioralModels > MixIMT_wrk > CKT_IMTA_extraction. Also see related component *MixIMTA_Setup (Single-RF Intermodulation Table Extractor)* (ccsys).

5. The internal topology of MixIMTB_Setup is generated dynamically during simulation. There is no viewable sub-circuit of this component. To understand the principle of IM table generation, see how the power sources and harmonic balance controller are set up for *MixIMTA_Setup (Single-RF Intermodulation Table Extractor)* (ccsys), which is more simple.

# MixIMT_Data (Multi-RF Intermodulation Table Mixer Data Model)

## Symbol
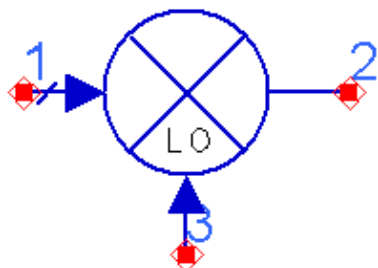


## Parameters

| Name | Description | Units | Default |
|---|---|---|---|
| Filename | Name of IMT file to be read | | "imtfile.imt" |
| RFharms[r] [†] | Number of harmonics of r-th RF input involved in mixing | None | 1, r=1 |
| RFfreq[r] [†] | Fundamental frequency of r-th RF tone | Hz | 1.1 GHz, r=1 |
| LOharms | Number of LO harmonics involved in mixing | None | 2 |
| LOfreq | Fundamental frequency of LO tone | Hz | 1.0 GHz |
| ZRF | Impedance of RF port | Ohm | polar(50,0) |
| ZIF | Impedance of IF port | Ohm | polar(50,0) |
| ZLO | Impedance of LO port | Ohm | polar(50,0) |
| ConvGain | Voltage conversion gain from RF to IF port | None | dbpolar(0,0) |
| RFreflection [††] | Wave variable reflection at RF port (S11) | None | polar(0,0) |
| IFtoRFleakage [††] | Wave variable leakage from IF to RF port (S12) | None | polar(0,0) |
| LOtoRFleakage [††] | Wave variable leakage from LO to RF port (S13) | None | polar(0,0) |
| RFtoIFleakage [††] | Wave variable leakage from RF to IF port (S21) | None | polar(0,0) |
| IFreflection [††] | Wave variable reflection at IF port (S22) | None | polar(0,0) |
| LOtoIFleakage [††] | Wave variable leakage from LO to IF port (S23) | None | polar(0,0) |
| RFtoLOleakage [††] | Wave variable leakage from RF to LO port (S31) | None | polar(0,0) |
| IFtoLOleakage [††] | Wave variable leakage from IF to LO port (S32) | None | polar(0,0) |
| LOreflection [††] | Wave variable reflection at LO port (S33) | None | polar(0,0) |
| [†] Add or remove elements of repeated parameters from dialog box. Ensure that identical number of harmonics and frequencies are specified at all times. [††] Reflection and leakage parameters apply to entire spectra present at relevant ports, although RF and LO ports are internally frequency sensitive. See note 6. | | | |

## Notes/Equations

1. MixIMT_Data is a behavioral mixer component that relies on IMT files. When supplied with an IMT file in O, A or B formats, it emulates mixer behavior by sensing power levels at specified frequencies on the RF and LO ports and reproducing the IF spectrum described in the data file at the output port. Based on the use of the RFfreq[] parameter, it is capable of sensing multiple unrelated fundamental tones at the RF input and generating intermodulation products as specified by multi-tone spurs on file. It senses only the LOfreq tone at the LO input and uses it for determining mixing products.

2. The basic behavior described above can be further modified by specifying non-default voltage conversion gain and port leakage/reflection values. See *Working with Data Files* (cktsim) > IMT Format for more information.

3. MixIMT_Data operates on each type of IMT file as follows:
   When supplied with an O-type (single-RF, single side banded data with no phase information) file, it generates identical upper and lower side band behavior for any mixing product. When supplying this type of IMT file, the user should ensure that the instance is made sensitive to only the one RF tone of interest. Note that these IMT files do not contain any frequency information. Thus the spur behavior described on file can be reproduced at any RFfreq and LOfreq of interest.
   When supplied with an A-type (single-RF, double side banded data with phase information) file, it generates distinct upper and lower side band behaviors as recorded on file. When supplying this type of IMT file, the user should ensure that the instance is made sensitive to only the one RF tone of interest. Since A-type IMT files contain actual values of frequencies and powers, if the RFfreq[1] or LOfreq specified on the behavioral instance does not match the file values, then frequency and power domain interpolation is done to predict a mathematically supportable behavior.
   When supplied with a B-type (multi-RF, double side banded data with phase information) file, it generates distinct upper and lower side band behaviors as recorded on file. When supplying this type of IMT file, the user should ensure that the instance is made sensitive to as many RF tones as are present on file. Since B-type IMT files contain actual values of RF frequencies, deviation of the RFfreq[r] from the behavioral instance of file values is not tolerated. Deviation of sensed RF power is allowed. Deviations of LO frequency/power and RF power at established frequencies are responded to by frequency and power domain interpolation as explained in *Working with Data Files* (cktsim) > IMT Format.

4. *IMT matrices up to [( 2\* RFharms[1] + 1 )\*..... \*( 2\* RFharms[r] + 1 )] x ( LOharms + 1) can be supported by this data model if a table is available on file. Although there is no theoretical upper limit on the number of harmonic indices that can be specified on the data model, the user should establish a cut-off on the number of IF spurs to improve the performance of the underlying harmonic balance simulation. Using the equation for matrix size defined above, it can be verified that a simple three-tone RF input, such as used for adjacent channel power rejection (ACPR) tests, mixed at RFharm[] = {3,2,2} with LOharm = 5 will generate a 175 x 5 IMT matrix. Lowering the spur precision by 1 on each RF tone will produce a more manageable 45 x 5 IMT matrix.*

5. To understand the use model for this data model see example designs: BehavioralModels > MixIMT_wrk > BEH_IMT_*. Also see related extraction components *MixIMTA_Setup (Single-RF Intermodulation Table Extractor)* (ccsys) and *MixIMTB_Setup (Multi-RF Intermodulation Table Extractor)* (ccsys).

6. The internal topology of MixIMT_Data is dynamically generated during simulation. It

is possible to trade off the number of spurs generated by the model against simulation performance. Compare resource consumption of example designs BEH_IMT_1R2R_2L and BEH_IMT_1R1R_2L from the example in Note #5. Both designs contain behavioral mixers which read the same data file. However, in the second case the desired degree of distortion is reduced on the second RF tone, resulting in faster simulation without sacrificing accuracy of any desired intermodulation product.

7. The MixIMT_Data model is based on an FDD which responds to exact frequencies of interest supplied on the model instance. When designing a mixer where colliding tones exist at the output spectrum, offset the frequency of one of the input tones by a negligible amount to ensure proper simulation. For example when signal frequency is RFfreq[1] = 2.0 GHz and image frequency is RFfreq[2] = 1.4 GHz, with LOfreq = 1.7 GHz, colliding tones are expected at the difference frequency of 300 MHz. Set RFfreq[2] = 1.4 GHz + 1e-3 Hz on the data model, the source, as well as the simulator if applicable. See design BEH_IMT_1R2R_2L in example available at BehavioralModels > MixIMT_wrk.

8. Applying conversion gain to a system level IMT mixer model is a means of altering the part of the IF spectrum at the IF port that is influenced by RF signal power, without affecting other tones or any other signals at any other ports. Given signal magnitude (in dBm) of RFdBm and phase RFdeg at the RF input, a conversion gain of value dbpolar (CGdB, CGdg) ensures that the internal RF magnitude (in dBm) is altered to RFdBm + CGdB and internal RF phase to RFdeg + CGdg prior to the mixing process without any impact on the external RF pin. The internal RF magnitude (in dBm) and phase is then used to generate the IF spectrum after appropriate scaling with respect to nominal RF power in the IM table. See design BEH_IMT_2R_4L_ConvGain in example workspace BehavioralModels > MixIMT_wrk.
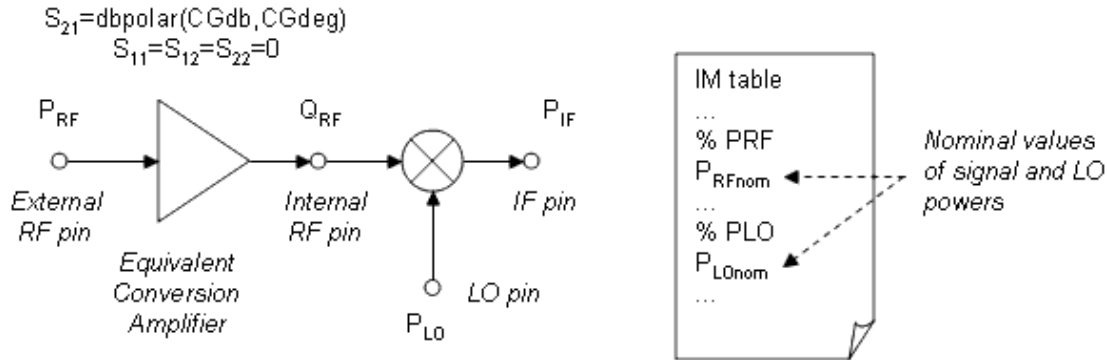


$$Q_{RF} = P_{RF} + \text{dbpolar(CGdb,CGdeg)}$$

Estimation of the change in the IF spur at nLO X mRF due to the supplied signal and conversion gain is computed as $\Delta P_{IF}$ such that:

$$dB(\Delta P_{IF}\{n_{LO}, m_{RF}\}) = |m_{RF}| * (P_{RF} + CGdb - P_{RFnom}) + |nLO| * (P_{LO} - P_{LOnom})$$

$$\text{phase}(\Delta P_{IF}\{n_{LO}, m_{RF}\}) = m_{RF} * (CGdeg)$$

> ℹ️ **Note** If there are phase shifts in signals applied at external RF or LO ports, they will impact the phase of the IF spurs in addition to the shifts introduced by the conversion gain parameter.

9. MixIMT_Data accepts all IMT values as representative of absolute magnitude (in dBm) (and phases when applicable) of IF spurs. Therefore, when trying to replace a MixerIMT2 model which was in use in the relative magnitude (in dBm) or dB mode, the corresponding IMT table should be manually converted into a DBM style table. The simplest way to achieve this is to manually edit all IF spurs by adding the

amount of the baseline magnitude (in dBm). There are two conventions for baseline power: one being the value of RF signal magnitude (in dBm) at RF input and the other being an expected IF fundamental magnitude (in dBm) at IF port. To address the former convention, the RF signal value is always included in an IMT file and could be applied as an additive to all spurs. If the latter convention is desired, then the absolute value of IF fundamental should be determined and applied to the Mix(1,1) data point in the table. All other spurs will need to be upgraded by the same amount.

# VCA_Data (Voltage Controlled Amplifier)

## Symbol
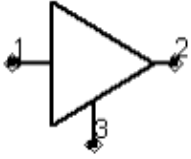


## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Dataset | Name of dataset generated by VCA_Setup model | None | dataset.ds |
| InstrumentID | ID of VCA_Setup extractor used to generate dataset | None | X1 |
| iVarN † | Name of Nth multi-dimensional variables N={1,...,N}; use if present in P2D file | None | None |
| iValN † | Value of Nth multi-dimensional variables N={1,...,N}; use real or integer values only | None | None |
| † If multi-dimensional variables are present in the Dataset, then these variables are required to be listed in iVarN as indicated in note 11 and corresponding values assigned to them using the iVaN parameter.Multidimensional VCA datasets are generated using parametric sweeps over the VCA_Setup component. | | | |

## Notes/Equations

1. VCA_Data is a data-based system model of a circuit-level VCA. The circuit level VCA is characterized by a dataset generated by the system component VCA_Setup. The use of this component is highlighted in the example *VCA_wrk.*

2. VCA_Data is the most advanced data-based system amplifier. If no special circumstances make AmplifierS2D, AmplifierP2D, AmpH1H2 or AmpLoadPull superior to VCA_Data, the latter model should be used for data-based system modeling of amplifiers. Since the VCA_Data does not have provisions for explicit control of frequency, order of harmonics or bias voltage at the instance level, it is advised that behavioral simulation and extraction simulation environments be closely matched for reliable modeling.

3. VCA_Setup and VCA_Data work equally well for amplifiers with or without a control voltage. For amplifiers without a control voltage, the control pin should be grounded.

4. VCA_Data is designed for Harmonic Balance or Circuit Envelope system verification (ACPR, BER, etc.) where various filters typically eliminate unwanted frequency components. Therefore, VCA_Data makes no attempt to model harmonic components. This means that for a 1-tone Harmonic Balance simulation at frequency f, only the frequency component at f (fundamental) is retained. All others will be invalid or zero as they are assumed to be filtered later.

For a 2-tone Harmonic Balance simulation at frequencies f1 and f2 (practically very close), only the frequency components at f1 and f2 (fundamentals) and those at 2 x f1-f2 and 2 x f2-f1 (intermodulation) are retained. These frequency components are very close and filtering cannot be guaranteed. All other frequency components are invalid or zero, it is assumed that they will be filtered later. Similarly, for a Circuit Envelope simulation, narrowband modulated components around the carrier(s) are retained. All others are invalid or zero, it is assumed they will be filtered later. VCA_Data will also run in *Transient* but this is not a recommended simulation controller. Since VCA_Setup is based on Harmonic Balance analysis, the dataset created by VCA_Setup contains steady-state information only. Transient analysis can only predict the correct steady-state response. This likely to result in an incorrect transient response.

5. In order for VCA_Data to produce an accurate model of the circuit-level VCA characterized via VCA_Setup, the Order parameter for VCA_Setup must be large enough to prevent aliasing of higher-order frequency components. For mildly non-linear circuits, an Order of approximately 5 should suffice while a range of 10-15 is recommended for highly non-linear circuits. Provided this criterion is met, the Order parameter for the simulation controller can be very low when using VCA_Data in a Harmonic Balance or Circuit Envelope simulation. Typically, an Order of approximately 3 should suffice for accurate VCA modeling. However, accurate modeling of the other components in the complete system may necessitate the need for a larger value for this parameter.

6. For a circuit-level VCA, the impedances presented by the input, output and control pins are usually highly complicated functions of all state variables within the circuit of which the VCA is part. For a data-based VCA, a simplification is necessary because no information about the environment in which the VCA will be placed is known at the time of extraction. Specifically, VCA_Data presents an input power and control voltage dependent impedance at the input, output and control pins, with the input and control impedances assuming an open-circuit output pin. The input and output impedances are voltage-to-current ratios at RF while the control impedance is a voltage-to-current ratio at DC.

7. Assuming proper sampling, the interpolation properties of VCA_Data are generally good, but occasionally break down near the limits of the ranges to which input power and control voltage were constrained when generating the dataset using VCA_Setup. To safe-guard against such a breakdown, the upper and lower limits of these ranges can be extended about 6 dB in input power and 1 V in control voltage.

8. The extrapolation properties of VCA_Data are very poor. When using VCA_Data, please ensure that input power and control voltage are within the ranges to which input power and control voltage were constrained when generating the dataset using VCA_Setup.

9. VCA_Data does not model noise.

10. VCA_Setup and VCA_Data can be used for simulating an amplifier with an arbitrary swept control parameter, for example, a temperature. Simply convert a 2-pin amplifier dependent on a given parameter, in this case Par, to a 3-pin amplifier with the third pin connected to a grounded DC voltage source whose strength equals Par. By sweeping the control voltage of VCA_Setup in the desired parameter range for Par and subsequently connecting the control pin of VCA_Data to a grounded DC voltage source, Par values in the appropriate range can be selected by setting the strength of this voltage source properly.

11. The response of VCA_Data can be user-selected from a body of a multi-dimensional VCA dataset by specifying the names and values of such variables in the *iVarN* and *iValN* parameters prior to behavioral simulation. Details of multi-dimensional VCA

dataset generation are discussed in *Notes/Equations* (ccsys) in the VCA_Setup component documentation. If a given multi-dimensional VCA dataset contains "V" variables in the form of external sweeps above the Vcontrol sweep, then the names of these variables need to be assigned to an iV *arN* parameter of the VCA_Data instance. Although the ordering of these names can be arbitrary relative to the order in the P2D file, the existence and ordering of values in the i *Val1 through iValV* parameters should always correspond to those in *iVar1 through iVarV* . If a multi-dimensional variable contained in the VCA dataset is not specified in the iVarN of a VCA_Data instance reading that file, then the simulation will be terminated due to insufficient data for multi-dimensional data indexing. Only real and integer values of multidimensional *iVarN* are supported.

12. A number of new features have been added to enhance the performance of VCA_Data in the ADS2004A release. Please note that the ADS2004A version of VCA_Data is compatible only with datasets generated by the ADS2004A version of VCA_Setup. If a VCA dataset generated by an earlier version of the extractor model is applied to a new version of the data model, an error message is generated asking the user to re-perform the extraction process prior to behavioral modeling.

# VCA_Setup (Voltage Controlled Amplifier Setup)

## Symbol
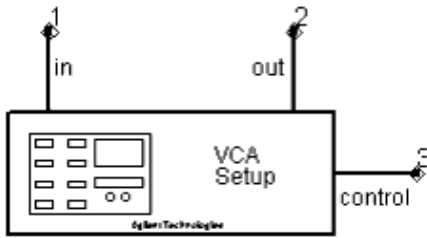


## Parameters

| Name | Description | Units | Default |
|------|-------------|-------|---------|
| Freq | Fundamental frequency; must be a non-negative real number | GHz | 1.0 |
| Order | Order for Harmonic Balance simulation inside extractor; must be an integer greater than 0 | None | 10 |
| Pin_Start | Start value for sweep of large-signal input power; must be a real number | dBm | $-50.0$ |
| Pin_Stop | Stop value for sweep of large-signal input power; must be a real number | dBm | 20.0 |
| Pin_Step | Interval for linear sweep of large-signal input power; must be a positive real number | dB | 10.0 |
| Vcontrol_Start | Start value for sweep of control voltage; must be a real number | V | 0.0 |
| Vcontrol_Stop | Stop value for sweep of control voltage; must be a real number | V | 1.0 |
| Vcontrol_Step | Interval for linear sweep of control voltage; must be a real number | V | 0.1 |

## Notes/Equations

1. VCA_Setup performs a swept control voltage and swept input power simulation of a circuit-level VCA and generates a dataset for subsequent use by the data-based system model VCA_Data. The use of this component is highlighted in the example *VCA_wrk* .
2. VCA_Setup and VCA_Data work equally well for amplifiers with or without a control voltage. For amplifiers without a control voltage, the control pin should be grounded.
3. In order for VCA_Data to produce an accurate model of the circuit-level VCA characterized via VCA_Setup, the Order parameter for VCA_Setup must be large enough to prevent aliasing of higher-order frequency components. For mildly non-linear circuits, an Order of approximately 5 is sufficient while an Order in the range 10-15 is recommended for highly non-linear circuits.
4. Push into VCA_Setup in a Schematic window for a view of the swept Harmonic Balance simulation controllers. This component can be copied and modified to suit individual needs; typically, this would entail changing the Harmonic Balance controller *HB1* to achieve more efficient simulations. For more information, refer to *Harmonic Balance Simulation* (cktsimhb).
5. When simulating an existing ADS2002 design containing a VCA_Setup instance in the current version of ADS, you must substitute a newer version of VCA_Setup. Failure to

do this will result in the generation of warning messages and corruption of the extracted dataset.

6. The extracted ADS dataset is assigned the name of the extraction design by default.

7. It is possible to generate arbitrary multi-dimensional VCA datasets using a VCA_Setup component by applying a chain of ParamSweep controllers in the extraction level design and assigning the SimInstance[1] value of the inner-most sweep to be "Xi.Vcontrol" where "Xi" is the instance name of the VCA_Setup component in use. *Vcontrol* is a default uppermost sweep inside the extractor subcircuit, hence this assignment bridges the gap between the minimum viable parameters necessary for voltage controlled amplifier modeling and those necessary to adapt such a model for your own application needs. Refer to the example design CKT_VCA_extraction for an understanding of the multi-dimensional extraction process. The datasets generated by ADS2003C and prior versions of VCA_Setup are significantly different from that generated by the ADS2004A and subsequent versions. To learn how to access the customized multi-dimensional VCA dataset refer to *VCA_Data (Voltage Controlled Amplifier)* (ccsys).